A New Process Model for the Comprehensive Management of Machine Learning Models

Christian Weber[1], Pascal Hirmer[2], Peter Reimann[1] and Holger Schwarz[2]
[1]*Graduate School advanced Manufacturing Engineering, University of Stuttgart, Nobelstraße 12, 70569 Stuttgart, Germany*
[2]*Institute for Parallel and Distributed Systems, University of Stuttgart, 70569 Stuttgart, Germany*

Keywords: Model Management, Machine Learning, Analytics Process.

Abstract: The management of machine learning models is an extremely challenging task. Hundreds of prototypical models are being built and just a few are mature enough to be deployed into operational enterprise information systems. The lifecycle of a model includes an experimental phase in which a model is planned, built and tested. After that, the model enters the operational phase that includes deploying, using, and retiring it. The experimental phase is well known through established process models like CRISP-DM or KDD. However, these models do not detail on the interaction between the experimental and the operational phase of machine learning models. In this paper, we provide a new process model to show the interaction points of the experimental and operational phase of a machine learning model. For each step of our process, we discuss according functions which are relevant to managing machine learning models.

1 INTRODUCTION

Companies increasingly try to automate decisions based on data and analytics (Lueth et al., 2016). Thereby, they encounter a multitude of diverse analytical applications that are often based on complex algorithms and highly scalable processing architectures. In such advanced analytical applications, the central artifact is a machine learning model, which is the result of applying a machine learning algorithm to data. The model is then used to infer predictions for new, unseen data to provide actionable insights that facilitate manual or automated decision-making. Thereby, the prediction performance of a model, e. g., its accuracy, is of paramount importance, since false predictions may lead to wrong decisions, which in turn may have a negative impact on the company's performance. Altogether, this turns models into valuable corporate assets that have to be managed actively (Krensky and Hare, 2017). This not only requires to manage the models themselves, but also their lifecycle and how the models evolve within individual phases of this lifecycle.

Related work proposes several process models providing guidelines for the steps to conduct analytical applications, e. g., the KDD-Process (Fayyad et al., 1996) or CRISP-DM (Chapman et al., 2000). The usual lifecycle phases a machine learning model encounters in these process models are (1) planning the model, (2) building and testing it, (3) deploying it in the real environment, (4) using it for predictions, and (5) retiring it after some time. While many process models support loops between the first two phases of planning and building models, they often consider the remaining phases as strictly sequential. However, real applications also require additional loops, especially from the phase of using models back to other phases. These loops starting from the usage phase entail a high complexity for managing models, different model versions, and their prediction performance.

For instance, a crucial management function is to maintain the models while they are in use within real and ever-changing environments. A good example are manufacturing environments, which are characterized by dynamics of manufacturing systems (Wuest et al., 2016). Dynamics like tool wear, parameter settings of machines, and different materials introduce changes in data and data distributions, which may result in false predictions of models. Thus, models have to be re-built and sometimes re-engineered in sophisticated ways to adapt to such situations.

In this paper, we introduce a new process model that supports the overall lifecycle of machine learning models in a comprehensive way and that especially considers complex update and maintenance loops as

Weber, C., Hirmer, P., Reimann, P. and Schwarz, H.
A New Process Model for the Comprehensive Management of Machine Learning Models.
DOI: 10.5220/0007725304150422
In *Proceedings of the 21st International Conference on Enterprise Information Systems (ICEIS 2019)*, pages 415-422
ISBN: 978-989-758-372-8

first-class citizens. We explain the individual steps of this process model and discuss considerations on related tasks for managing and maintaining machine learning models.

The remainder of this paper is organized as follows. First, we present a motivating scenario to show the need for a new process model in Section 2. In Section 3, we discuss related work about process models for generating and operating machine learning models. Next, we introduce our concept for a new process model supporting the lifecycle of machine learning models in Section 4. In Section 5, we summarize the paper and discuss future work.

2 MOTIVATIONAL SCENARIO AND RELATED CHALLENGES

In this section, we discuss a motivating scenario from the manufacturing industry in which model management plays a crucial role (Brenner et al., 2018). We highlight problems in the supporting process of managing the underlying machine learning models. In a simplistic scenario the model is planned, built, tested, used, deployed, maintained and, after some time, retired. Participants in this process are the business unit which delivers domain knowledge for constructing machine learning models, a central analytics unit which builds the analytical solutions and the IT unit which integrates and maintains these solutions.

2.1 Scenario

Plan Model. In our manufacturing scenario machines produce scrap material due to a complex combination of materials and parameter settings for machines which are not adequate. During the production process a lot of errors are introduced by tool-wear. Some experienced maintenance workers can detect some but not all settings that result in problems. Often, the same problems occur but they are not detected in time. Therefore, the business unit requests a data-driven solution from the analytics unit to detect possible issues in time. The goal is to reduce the errors in the process through predictive maintenance. The business unit has a lot of data from past production schedules which they provide to the analytics unit. The result of discussions between the Business, IT and Analytics unit is to build a smartphone app for the maintenance workers that provides the maintenance orders in time. In another plant, a similiar solution has already been developed. As this is unknown to the data scientists they start a complete re-development.

Build & Test Model. After the data scientists of the analytics unit acquired domain knowledge via workshops with the IT and Business unit, they start working on the model. Often it takes a long time to understand and prepare the data for model training. After some time working on different models, they detect a promising combination of feature engineering, algorithms and hyperparameters. Later, after hitting a dead-end with some algorithm, they want to re-create another promising experiment they conducted some weeks ago. However, they cannot remember the exact experimental setup and start to build the model from scratch. Some results show a good accuracy. These promising models are deployed for testing and after some time they select the most appropriate model. Finally, data scientists hand over the final scripts and a model file to the IT department.

Deploy Model. The IT unit integrates the model into the webservice and builds the smartphone app. For this task, they have to reimplement the scripts, because the model format is not supported by libraries for the target programming language. A lot of communication efforts are needed between the IT and the analytics unit to transfer knowledge about building the model.

Use Model. A long time elapses between the provisioning of the data and the receipt of the solution. The model is delivered as a smartphone app which monitors anomalies in the process. With the app, maintenance workers handle maintenance jobs in time, preventing down-times and scrap material. But soon, workers experience that tools are changed for new ones, even though these have no significant wear at all. After some time, the workers do not trust the model results anymore. The model lost its accuracy and produces wrong predictions. Something in the data has changed. Some time later, a technician diagnoses that there is a defective temperature sensor which delivers false results, resulting in false predictions of the model. The sensor is replaced with a new one and the model delivers again accurate predictions. Two weeks later, the model once again produces false predictions. The technician checks the sensors, but he cannot detect any problem. The business unit contacts the analytics unit to request maintenance for the model. The analytics unit knows that changing circumstances in the production environment are responsible for the degraded model. They refer the business unit to the IT unit because they do not see model maintenance as their responsibility. Instead of searching for the root-cause of changed data, the IT unit decides that it is easier to adapt the model to changes in the data. The IT experts are confused because knowledge about the concepts of updating these models is

far beyond their knowledge (Fayyad et al., 2017). For that reason, they contact the analytics unit and more communication efforts are necessary. The business unit is annoyed by long reaction times of the IT and the analytics unit. Especially because it is unclear which unit is responsible for maintaining the model. After some time, the business unit decides to revert to the old manual solution.

Retire Model. Due to missing knowledge and high operating costs, models get retired before finding an appropriate maintenance strategy. This is often the case, when knowledge of the overall process of developing and operating a model is unknown to the involved departments. This leads to high expenditure of time and money, missunderstandings and a loss of opportunities. In the next section, we detail some of these problems.

2.2 Need for Update/Upgrade Loops in Process Models

Building a machine learning model is considered to be explorative and ad-hoc. Data scientists try many combinations of data preprocessing, algorithm selection, hyperparameter tuning and testing to produce a model with the desired performance. Due to the high amount of iterations, we consider that loop as *experimental loop*. This loop has recently received much attention from the scientific community regarding functions for managing models (Miao et al., 2017; Schelter et al., 2017; Vartak, 2017). However, these approaches do not focus on loops that origin from the operational phase. For example, when a model is deployed and in use, data can change and the model degrades after some time. This is considered to be a concept drift (Gama et al., 2014). Concept drifts can have various reasons. In industrial environments where predictions are applied to streams of sensor data, sensors of machines can be defective or deliver false values because of tool wear. Our motivational scenario shows that the root cause for these issues is often not detected in time or not detected at all. Thus, a solution is to adapt the model to the changed situation. This can be achieved by re-training the model with more recent data. This means that the model is rebuilt, but the preprocessing steps for the data, the algorithm and the hyperparameters remain fixed. The old version of the model is replaced by the new one. This can happen multiple times, resulting in what we call an *update-loop*. However, if update-loops do not yield the desired accuracy, more profound changes must be made. For example, additional data and different preprocessing steps have to be tested with a different algorithm, leading to a new variant of the model for the same use case. The model is re-engineered. We consider this as an *upgrade loop* of the model. We think it is important to also consider the upgrade and update loops in the overall lifecycle of a model and according functions for model management, because stakeholders need to base decisions on these loops. For example, they have to decide for or against update/upgrade strategies, accountability of stakeholders for maintenance, supporting tools and economic viability of the model.

3 RELATED WORK

In this section, we discuss related work about process models for the lifecycle of machine learning models. Existing and more general process models for data mining projects are the KDD process (Fayyad et al., 1996) and CRISP-DM (Chapman et al., 2000). The KDD process describes various steps that have to be carried out to generate knowledge from data. The majority of steps describe preprocessing activities to select, clean, and transform input data. Afterwards, the actual data mining takes place where interesting and previously unknown patterns are derived. These patterns may be represented as mining or machine learning models that are descriptive, predictive, or a combination of both. In a last step, the patterns and the models are interpreted to gain valuable knowledge regarding the relevant domain-specific problem. The KDD process supports various loops among all these steps. However, it does not give hints on how to deploy and subsequently use the resulting machine learning models within a real environment, e. g., for making predictions when machines may fail in a manufacturing environment. Consequently, it does not describe how to conduct loops for upgrading or maintaining the models that are already in the usage phase.

CRISP-DM treats the analysis of the business problem, the investigation and preparation of data, as well as building and testing machine learning models as core process steps. Unlike the KDD process, CRISP-DM explicitly considers a step, where machine learning models are deployed within the real environment. However, it likewise does not cover a usage phase and corresponding upgrade or maintenance loops from this usage phase back to the other steps of the overall process. In CRISP-DM, knowledge about the overall data mining process is documentated via reports at the end of each step. It thereby also covers monitoring and maintenance plans for updating models. Such a plan specifies which models require updating and why. Furthermore, it defines the trigger of the update, e. g., regular updates, trigger events, or per-

formance monitoring. However, the contents are not further detailed, nor reflected in the process model.

A more recent approach is the Team Data Science Process (TDSP) of Microsoft (Ericson et al., 2017). This process also includes a deployment step in which a pipeline is built for the update/upgrade of models. In comparison to our process, the update and upgrade loops are not depicted as loops in the process model. Instead, these are covered in pipeline development.

The most similiar approach that inspired our process model is the lifecycle of an analytical model provided by (Grossman, 2018; Pivarski et al., 2016). They provide a phase of analytical modeling and an analytic operations phase in the development and operation of machine learning models. For the update and upgrade of models they describe a champion/challenger strategy. This strategy is covered by a patent of (Chengwen, 2012). It contains a parallel process step, which includes building other models versions, while the current version of the model is in use. If the new version of the model is more accurate, it will replace the old one. However, the champion/challenger strategy is not a particular strategy for triggering the update of a model. Triggering updates can be achieved through, e.g., manual or periodic retraining and adaptive machine learning algorithms to react to concept drifts. The final step is to retire the model and to re-deploy an improved one. In our process, we consider retiring the model as a final activity when the model is taken out of service. We consider a model to consist of multiple model versions that represent the evolution of the model over time. These model versions are generated through our introduced update and upgrade loops.

4 PROCESS MODEL

In this section, we introduce our main contribution, a new process model for comprehensive machine learning model management. Loops need to be added to the process in order to enable reaction on changes in either the model or in the context the model is applied to. The process is divided into two phases: (i) the experimental phase, in which models are planned, built, and tested, and (ii) the operational phase, in which models are used, monitored, and, if necessary, re-built or retired. This is depicted in Figure 1. In a first step, we conducted a thorough analysis in the smart manufacturing domain depicted by the motivational scenario (see Section 2), in order to derive essential features and extensions of the process. First, it is necessary to enhance the processed models with context information, which gives further insights on how they are used and whether they need to be updated to fit their context. Second, stake-holder specific requirements need to be considered. Depending on the domain, these requirements can be very heterogeneous. However, they are of great importance to tailor the process to the specific needs of the use cases. In the following, we describe the steps of our process, focusing on the update and upgrade loops, as highlighted in Figure 1. For each step we describe relevant functions to manage machine learning models.

4.1 Step 1: Plan Model

In this step, it is important to consider specific requirements by the corresponding use case, its domain, and the involved stakeholders. Making mistakes in the model planning step can lead to costly re-planning and to misleading results. Consequently, business requirements need to be defined first, considering the goals of the desired use cases. Second, stakeholder workshops need to be organized, in which all involved stakeholders can verify and evaluate these requirements and, if necessary, change them or come up with new ones. Furthermore, it needs to be evaluated whether the desired model is feasible, i.e., whether it is economical, it reaches the goals that need to be solved and if the necessary data are available. The planning step is essential for the (economical) success of the model. If the desired model is not feasible, for example, because the required data is not available or the costs for realization are too high, it is discarded as depicted in Figure 1 after Step 1. Consequently, the process directly moves to the retirement of the model (cf. Figure 1, step 6).

Management Considerations. In this step, the model needs to be semantically enriched with planning data that is further refined in the upcoming steps of our process model. The planning data is mainly related to business concerns and the expected usage of the model in production. Planning data include information about, e.g., the corresponding use case, the prediction that should be made, and the decision based on the model. A method that can be used to collect planning data via stakeholder workshops is the machine learning canvas (Dorard, 2019). It enables to align the domain knowledge of business people, data scientists and IT experts. However, the machine learning canvas is rather static and provided as a document template. It would be useful to store its contents and link them to models in order to enable a semantic search for models. For example, data scientists want to search models to get a rough idea on how to develop a new model for a similiar use case.

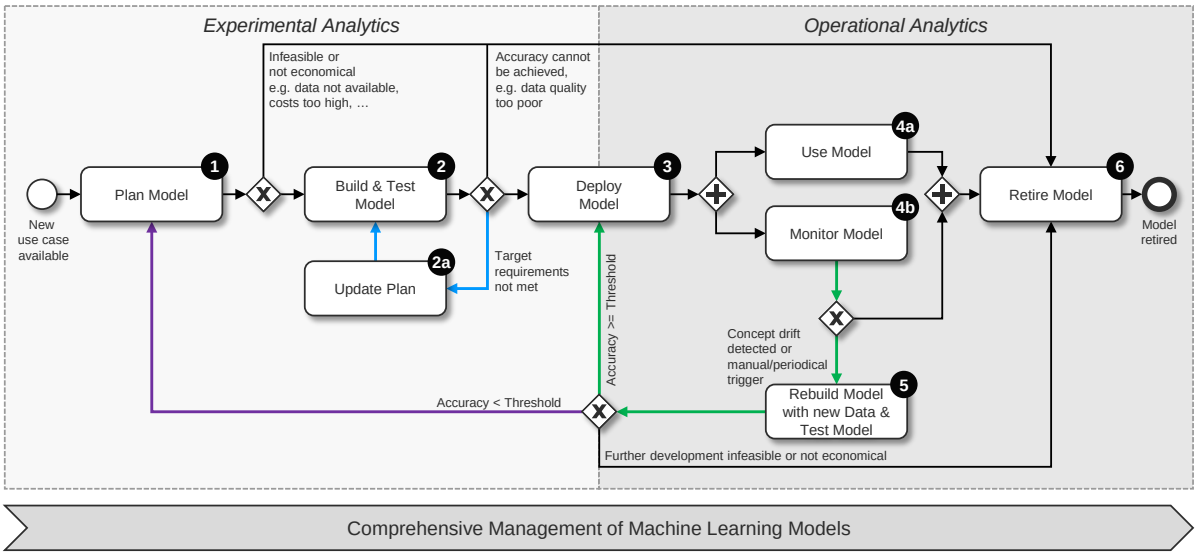

Figure 1: Process Model for the comprehensive management of machine learning models. Green: update loop, purple: upgrade loop, blue: experimental loop.

4.2 Step 2: Build and Test Model

In the next step (cf. Figure 1, step 2), a first version of the model is built and tested. Depending on the complexity and the needed processing power, this could be done either on the local machine or a distributed environment like Apache Spark[1]. Typically, models are built using script-based programing languages, such as Python or R, using available libraries for data cleaning, transformation, and machine learning. Another approach is using sophisticated tools, such as Rapidminer[2] or KNIME[3], that offer a wide variation of modules to process, transform, and analyze data. These tools can typically be used by business users without programing or IT experience due to a mostly graphical modeling approach. The result is comprised of a set of scripts, cleaned datasets, and serialized models. In this step, the model is also tested, e.g., it is verified if its quality reaches a desired threshold. If the quality is too poor and its quality cannot be increased, the model is discarded (cf. Figure 1, step 6). Furthermore, if the model does not meet the expected target requirements (Figure 1, step 1), it needs to be re-planned, e.g., by involving new data sources, selecting different preprocessing steps or trying different algorithms. This is conducted in a newly inserted step of our extended lifecycle process, the "Update Plan" step (cf. Figure 1, step 2a).

[1]https://spark.apache.org/
[2]https://rapidminer.com/
[3]https://www.knime.com/

Step 2a: Update Plan

Typically, model building is an explorative and iterative process and considered to be ad-hoc. This leads to the experimental loop in which step 2a is invoked if the built and tested model does not meet the requirements of the planning phase. In this step, the model needs to be improved according to the requirements determined in the planning step of the model lifecycle. This might also require additional input of the stakeholders to meet the desired requirements of specific use cases. After the improvements are identified, the model is re-planned and needs to be adapted using the above mentioned approaches, either tool-based or script-based. Each adaptation of the model can be regarded as an experiment through which a new model version is created. The model is represented as an object which can be serialized and saved to disk. Once the model is built, it needs to be thoroughly tested again regarding its quality and fulfillment of the target requirements it aims for. Step 2a is repeated until the model reaches the desired quality and is ready for deployment or, in the worst case, until it needs to be discarded due to poor quality. Note that having many iterations between Step 2 and Step 2a is costly and requires high effort. Consequently, the planning and building phase should be conducted thoroughly in order to reduce such costly re-planning and adaption steps.

Management Considerations. One challenge is to facilitate knowledge transfer from previous experiments in order to shorten iterations in the model building step. Recent supportive concepts that cope with

this issue can be found in (Miao et al., 2017; Schelter et al., 2017; Vartak, 2017). Often data scientists face a dead-end when they iterate on various prototypes of a model. For that reason they want to roll back to a previous promising experiment they had not pursued back then. Moreover, unsuccessful strategies from past experiments should not be repeated in future experiments. Due to hundreds of different experiments data scientists cannot remember all past experimental settings. These typically consist of a combination of feature engineering, algorithm selection and hyperparameters that lead to an experimental result (Kumar et al., 2015). In order to facilitate knowledge transfer from experiments, the result, the experimental settings, and the related artifacts like code, datasets and model files must be stored. This results in a need for tools like MLflow[4] to manage experiments.

4.3 Step 3: Deploy Model

This step is on the edge between the experimental phase and the operational phase of the process model. The goal is to deploy the built and tested model in its target environment (cf. Figure 1, 3). This is also referred to as model serving (Breck et al., 2017). This step presumes that the model fulfills the quality requirements and target requirements of the planning step. There are different possibilities how a model can be deployed. For example, it can be integrated into a database, run as a stand-alone application or it can be provided as a web service, offering its capabilities as a service. The deployment step can either be conducted manually, by setting up the business logic, the involved data, and the corresponding applications through an IT expert, or in an automated manner. Automated deployment of models and applications, respectively, can be realized through software provisioning approaches, such as Docker, or through the OASIS standard TOSCA (OASIS, 2013a; OASIS, 2013b). Often, multiple promising models are deployed for A/B testing, also referred to as split testing.

Management Considerations. In this step, the model is represented as a model file. It contains a serialized version of the final model object that was built in the previous step. For comprehensive model management one has to keep track of these model files and a function for dealing with model objects is required. Managing Model objects includes to keep track on deployed model objects and to provide means to compare and analyze them. Comparing and analyzing includes to identify differences between model objects. For example, one wants to identify which aspects in the model object changed after conducting two similiar experiments. This can become very challenging when the model is represented in heterogenous formats. In order to keep track on where models are deployed, metadata about deployment targets have to be stored as well.

[4]https://mlflow.org/

4.4 Steps 4a/b: Use & Monitor Model

Step 4 depicts the main step of the lifecycle process, the use and monitoring of the deployed models (cf. Figure 1, steps 4a and 4b). After a model is deployed either integrated in a database, an existing application, stand-alone or as a service, it is executed based on its corresponding business logic (e.g., realized through scripts or analytics tools). By comparing the results based on the test data, the performance can be evaluated. There are two approaches for predictions: (i) batch predictions, and (ii) realtime predictions. Batch predictions are done offline in batches while realtime predictions are done online during runtime. While the model is in use, its quality can decrease over time due to changes in the environment. More precisely, due to a concept drift, e.g., a change in the environment that has consequences on the model itself, the model's quality decreases. For example, this can be observed by monitoring changes in the distribution of input data that are fed into the model. Consequently, steps need to be taken in order to react to such occurring concept drifts. These steps lead to an update of the model (Figure 1, step 5) in order for it to be fitting to the changed context. The model is updated and re-deployed before it is used again.

Management Considerations. A model is updated when it degrades and an old model version is replaced with a new one. While the model is in use, it has to be ensured that a model produces accurate predictions. If predictions are found inaccurate, it should be possible to trace back which model version created these false predictions. Furthermore, when the model version is related to the according experiment, errors in the building step can be identified and avoided. For that reason, data across the process steps need to be linked. This allows to track provenance throughout the created artifacts in the experimental phase and the operational phase.

4.5 Step 5: Rebuild and Test Model

As mentioned in the previous step, a model may degrade over time and may not meet the accuracy it had before due to occurring concept drifts. There can can be two reasons for degrading. The first reason is that

the distribution of the data changes. A prominent example is a defect sensor in a machine, which produces false sensor values (see section 2). This is considered to be a virtual concept drift (Gama et al., 2014). The second reason is a real concept drift. This means that the distribution of the predicted value changes because of, e.g., a change of interest. To cope with virtual concept drifts, algorithms exist that conduct an automated concept drift detection. A second option is to build a more robust model by using ensemble techniques like bagging and boosting (Gepperth and Hammer, 2016). A further option is to periodically update the model, which is conducted in this step. After updating the model using new or changed data, it needs to be tested once again in order to ensure its accuracy. If the accuracy reaches an acceptable threshold (which is depending on the scenario), it can be directly deployed and used once again. However, if the re-built model cannot reach this threshold, for example, due to an extensive concept drift, it needs to be planned from the beginning.

Management Considerations. It is important to log the time needed for rebuilding the model so that it can be detected how much costs are involved and if further development is economical. For each rebuild one can save information about the training time and the accuracy of the updated model. By comparing the accuracy of the newly deployed model with the accuracy of previous versions of the model, we can recognize patterns for the degrading of models. For version control, every time a new model is built, it needs to be saved as a new model version. Also, when a completely new model is built, this model should be a variant to the original built model. Thus, the active management of models has to deal with model variants and models versions that emerge over time.

4.6 Step 6: Retire Model

The final step (Figure 1, step 6) represents the end of the model lifecycle process. It is reached if a model needs to be retired after building and testing due to economical issues, poor quality or if the model has been in use for some time and there is no more use for it, e.g., due to changed circumstances in the domain or due to finishing a project.

Management Considerations. When a model is retired, the according data and metadata of the model should be archived as reference for future experiments. Especially when legal regulations like the General Data Protection Regulation (GDPR) (European Parliament, 2016) require to justify past decisions based on the now retired model, archiving is crucial.

5 SUMMARY AND FUTURE WORK

In this paper, we introduced a new process model for the comprehensive management of machine learning models in order to show the importance of update and upgrade loops as well as according management functions. The result is a process that has its strengths especially in environments in which the context changes regularly and there is an increased probability that models degrade. This applies especially to the manufacturing domain, as described in Section 2. Our process helps to understand the required complexity for updating and upgrading models. This imposes new challenges for IT architects in implementing comprehensive model management as a business capability. In order to support these efforts, it is essential to conduct further research on blueprints for model management platforms, frameworks and tools. For this reason, we plan to integrate selected management functions into a platform for managing models in Industry 4.0 environments

ACKNOWLEDGEMENTS

The authors would like to thank the German Research Foundation for financial support of this project as part of the Graduate School of Excellence advanced Manufacturing Engineering at the University of Stuttgart. Some work presented in this paper was performed in the project 'MMP' as part of the Software Campus program, which is funded by the German Federal Ministry of Education and Research (BMBF) under Grant No.: 01IS17051.

REFERENCES

Breck, E., Cai, S., Nielsen, E., Salib, M., and Sculley, D. (2017). The ML Test Score: A Rubric for ML Production Readiness and Technical Debt Reduction. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 1123–1132.

Brenner, D., Weber, C., Lenz, J., and Westkaemper, E. (2018). Total Tool Cost of Ownership Indicator for Holistical Evaluations of Improvement Measures within the Cutting Tool Life Cycle. *Procedia CIRP*, 72:1404–1409.

Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., and Wirth, R. (2000). CRISP-DM 1.0: Step-by-step data mining guide.

Chengwen, R. C. (2012). Computer-implemented systems and methods for updating predictive models.

Dorard, L. (2019). The machine learning canvas.

Ericson, G., Rohm, W. A., Martens, J., Casey, C., Harvey, B., and Gilley, S. (2017). What is the Team Data Science Process?

European Parliament (27.04.2016). General Data Protection Regulation: GDPR.

Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996). The KDD Process for Extracting Useful Knowledge from Volumes of Data. *Communications of the ACM*, 39(11):27–34.

Fayyad, U. M., Candel, A., La Ariño de Rubia, E., Pafka, S., Chong, A., and Lee, J.-Y. (2017). Benchmarks and Process Management in Data Science: Will We Ever Get Over the Mess? In Matwin, S., Yu, S., and Farooq, F., editors, *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '17*, pages 31–32, New York, New York, USA. ACM Press.

Gama, J. a., Žliobaitė, I., Bifet, A., Pechenizkiy, M., and Bouchachia, A. (2014). A survey on concept drift adaptation. *ACM Comput. Surv.*, 46(4):44:1–44:37.

Gepperth, A. and Hammer, B. (2016). Incremental learning algorithms and applications. In *ESANN 2016 : Bruges, Belgium, April 27-28-29, 2016 : proceedings*, pages 357–368.

Grossman, R. L. (2018). A framework for evaluating the analytic maturity of an organization. *International Journal of Information Management*, 38(1):45–51.

Krensky, P. and Hare, J. (2017). Hype Cycle for Data Science and Machine Learning, 2017.

Kumar, A., McCann, R., Naughton, J., and Patel, J. M. (2015). Model Selection Management Systems: The Next Frontier of Advanced Analytics. *ACM SIGMOD Record*, 44(4):17–22.

Lueth, K. L., Patsioura, C., Williams, Z. D., and Kermani, Z. Z. (2016). Industrial Analytics 2016/2017: The current state of data analytics usage in industrial companies.

Miao, H., Chavan, A., and Deshpande, A. (2017). ProvDB: Lifecycle Management of Collaborative Analysis Workflows. In *Proceedings of the 2nd Workshop on Human-In-the-Loop Data Analytics*, HILDA'17, pages 7:1–7:6, New York, USA. ACM.

OASIS (2013a). Topology and Orchestration Specification for Cloud Applications.

OASIS (2013b). TOSCA Primer. Online.

Pivarski, J., Bennett, C., and Grossman, R. L. (2016). Deploying Analytics with the Portable Format for Analytics (PFA). In Krishnapuram, B., Shah, M., Smola, A., Aggarwal, C., Shen, D., and Rastogi, R., editors, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*, pages 579–588, New York, USA. ACM Press.

Schelter, S., Böse, J.-H., Kirschnick, J., Klein, T., and Seufert, S. (2017). Automatically Tracking Metadata and Provenance of Machine Learning Experiments. *Machine Learning Systems Workshop at NIPS*.

Vartak, M. (2017). MODELDB: A System for Machine Learning Model Management Chaminade, CA, USA, January 8-11, 2017, Online Proceedings. In *CIDR 2017, 8th Biennial Conference on Innovative Data Systems Research, Chaminade, CA, USA, January 8-11, 2017, Online Proceedings*. www.cidrdb.org.

Wuest, T., Weimer, D., Irgens, C., and Thoben, K.-D. (2016). Machine learning in manufacturing: Advantages, challenges, and applications. *Production & Manufacturing Research*, 4(1):23–45.