Online-Journals.org

iJIM International Journal of Interactive Mobile Technologies

iJIM | eISSN: 1865-7923 | Vol. 17 No. 23 (2023) | OPEN ACCESS

https://doi.org/10.3991/ijim.v17i23.43479

PAPER

Machine Learning Models Monitoring in MLOps Context: Metrics and Tools

Anas Bodor[1](✉), **Meriem Hnida**[1,2], **Najima Daoudi**[1,3]

[1]ITQAN Team, LyRica Lab, Information Sciences School, Rabat, Morocco

[2]RIME Team, Mohammadia School of Engineers, Mohammed V University, Rabat, Morocco

[3]SSLab, ENSIAS, Mohammed V University, Rabat, Morocco

anas.bodor@esi.ac.ma

ABSTRACT

In many machine learning projects, the lack of an effective monitoring system is a worrying issue. This leads to a series of challenges and risks that compromise the quality, reliability and sustainability of models deployed in production. As Machine Learning gains importance in various fields, poorly implemented monitoring represents a major obstacle to realizing its full potential. This article presents a comprehensive guide of machine learning models monitoring metrics and tool used in the MLOps context. The monitoring of metrics is important to evaluate and validate the performance of a machine-learning model, not only throughout the development phase but also during its deployment in the production environment. It enables real-time data to be collected on various metrics. The purpose of monitoring in MLOps context is to identify potential issues and adjustments made accordingly, guaranteeing consistent model quality and reliability. This article provides a comprehensive guide that introduces and explains a wide range of metrics used for continuous monitoring of ML systems at various stages of the MLOps lifecycle. Additionally, it presents a comparative analysis of available monitoring tools, enabling organizations to optimize their performance and ensure the seamless deployment of their machine learning applications. In essence, it underscores the critical importance of continuous monitoring and tailored metrics for ensuring the success and reliability of machine learning systems.

KEYWORDS

machine learning, MLOps, metrics, monitroring tools, continuous monitoring

1 INTRODUCTION

Machine learning (ML) has become an integral part of many fields, with its potential for improving performance and making the right decisions. Technology has seen growing adoption, fueled by increased computing power, more data, and algorithmic advances. However, to fully exploit the benefits of ML, it is essential to put in place robust industrialization processes. Industrialization enables ML models to be transformed into concrete products or services that can be integrated into existing systems and deployed on a large scale [27] [28] [29].

Bodor, A., Hnida, M., Daoudi, N. (2023). Machine Learning Models Monitoring in MLOps Context: Metrics and Tools. *International Journal of Interactive Mobile Technologies (iJIM)*, 17(23), pp. 125–139. https://doi.org/10.3991/ijim.v17i23.43479

Article submitted 2023-07-26. Revision uploaded 2023-10-22. Final acceptance 2023-10-22.

Despite significant advances in the field of artificial intelligence the rate of industrialization and deployment to production of ML projects is still low, and the transition of ML models from development stage to production is time consuming, depending on the requirements and particularities of each project. This is supported by the results of a survey carried out in 2020 by Sasu Makineth et al [1], which shows that professionals are concerned about the production and deployment of the various models developed. This can be explained by the fact that deploying ML models in production requires a cohesive integration of skills between software engineering and ML disciplines.

The need to standardize and unify the way in which an ML project is carried out has given rise to the MLOps paradigm, inspired by the field of software engineering. Taking advantage of the basic principles of DevOps, such as continuous integration (CI) and continuous deployment (CD), MLOps extends DevOps principles to the management of ML models, automating and optimizing their development, deployment, and maintenance in production environments. For instance, new concepts specific to MLOps have emerged: continuous training and experiment tracking. As researchers in artificial intelligence, our main research focus is on MLOPS. More specifically, our aim is to improve the efficiency and reliability of deploying ML models in large-scale production environments using MLOPS practices and tools.

This brings us back to the question of quality measurement and standardization, with the aim of making the overall environment of an information system dealing with ML programs comparable by attempting to define a unified model with a clear implementation architecture. To this end, several research projects have given rise to proprietary and open-source platforms and tools, such as Mlflow [3], Kubeflow [4], DVC [5] and many others, each of which attempts to handle and ensure one or more phases of the MLOps process.

The aim of this paper is to study monitoring as a transversal phase in the MLOps process, by (1) presenting challenges encountered during ML Model Deployment, (2) identifying and defining metrics for measuring the state of health of a model and, (3) presenting and comparing different tools used for continuous monitoring in the MLOps context. The scope of this approach encompasses all machine learning algorithms to be deployed in production, with the aim of ensuring continuous monitoring.

The reminder of this article is organized as follows: the second section presents the context and problem statement, with an overview of MLOps and its Lifecycle. The purpose is to contextualize monitoring within the overall production of ML models chain. The third section dives in depth into the monitoring process in an MLOps context, followed by the stages of monitoring a ML pipeline in the fourth section. In the fifth section, monitoring metrics and their categories are defined and explained. The sixth section provides a comparison of ML model monitoring tools. Finally, we summarize the key findings and highlight the crucial role of continuous monitoring in ensuring successful and sustainable ML model deployments.

2 CONTEXT AND PROBLEM STATEMENT

MLOps is the combination of two words: Machine Learning and Operations [6]. This combination represents the set of practices, tools and processes used to manage operations related to ML models, from development to deployment and even real-time monitoring of models in production. The challenges of continuous use of machine learning in applications date back to 2015, in a research paper entitled "Hidden Technical Debt in Machine Learning Systems"[16]. The authors in [22] [23] [24] defines MLOps as the iterative process of pushing the latest best ML models to production.

This paper aims to clarify and propose a global architecture to place monitoring as a transversal phase in the industrialization process of Machine Learning projects within the framework of an MLOps approach. This paper represents the state of the art in scientific research concerning this important component that ensures the quality of an ML model to continue delivering very good results over time.

2.1 Overview of MLOps and lifecycle

Inspired by DevOps principles, the MLOps paradigm includes the concept of continuous training (CT), to which we add continuous integration (CI) and continuous delivery (CD). The need to continuously monitor the health of models in production has given rise to a new concept called: Continuous Monitoring (CM) [2].

In this context, it is essential that the principles and concepts of continuous x (by x we mean CI/CD/CM) in MLOps are clearly manifested throughout the lifecycle of an ML project, as will be illustrated in the following part of this current section.

The practice of MLOps involves a series of steps aimed at overcoming the challenges associated with industrializing ML projects Following an extensive analysis of numerous scientific articles, we have been able to develop a global perspective on the MLOps lifecycle. Figure 1 illustrates the various stages. The aim is to better manage large-scale ML projects and deploy them effectively in a production environment [8]. By following these steps, professionals can better understand the different phases of the ML project lifecycle, including development, deployment, monitoring, and maintenance.

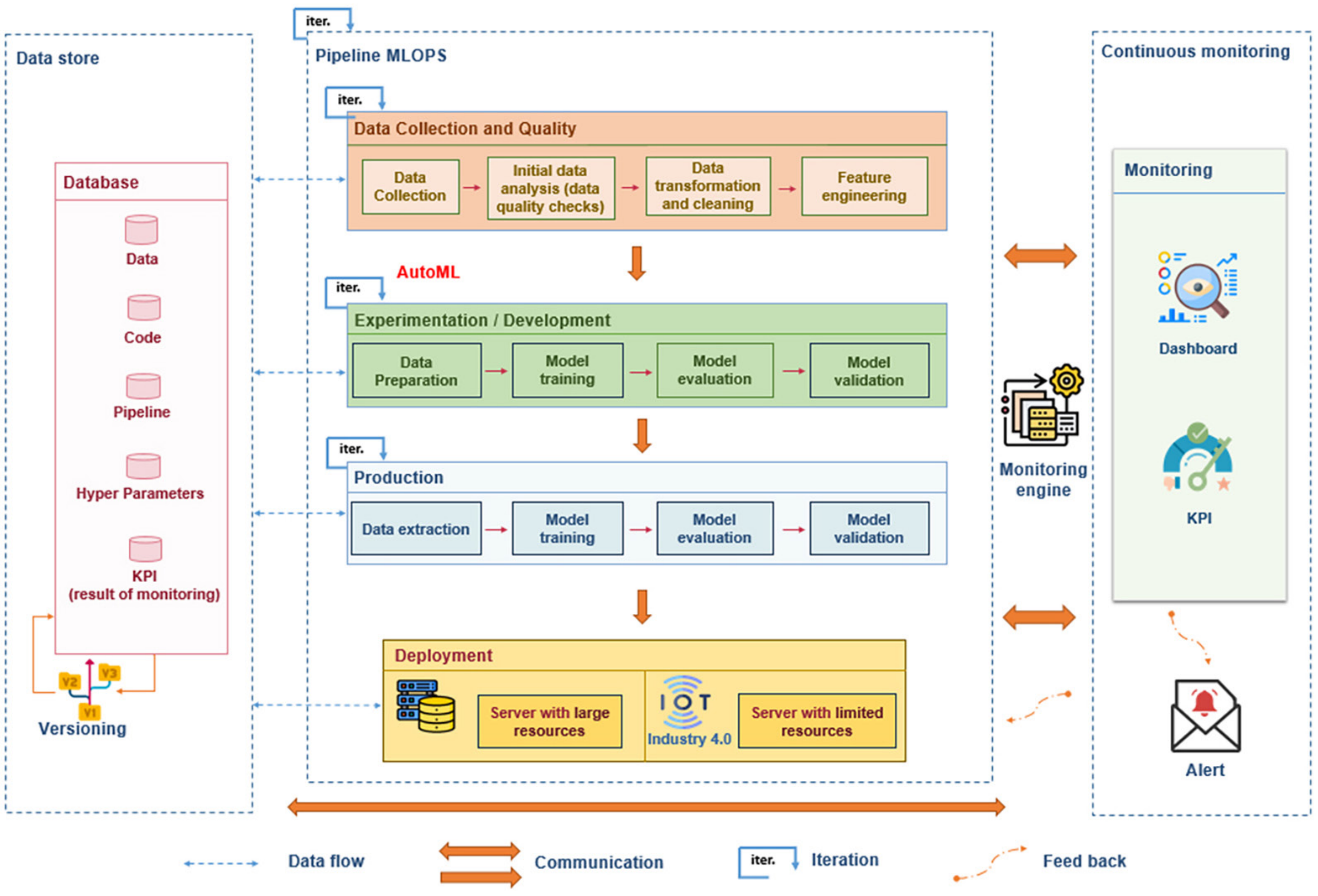


Fig. 1. Machine learning project lifecycle within an MLOps approach with a focus on the monitoring phase

The development process of ML project, also known as the pipeline [2], comprises a series of steps that are both linear and iterative (block Pipeline MLOps Figure 1). The pipeline begins with data extraction, validation, and preparation, followed by model training, evaluation and validation [31]. Once the model has achieved satisfactory results, deployment, and integration of the model into the final system can take place. These steps are essential to ensure that the program functions correctly in its production environment and can be effectively maintained over the long term.

Another aspect which is essential to the success of an ML project is the collection of metadata linked to each model drive, based on versioning [15] of data (block Data Store Figure 1), code, hyper-parameters, pipeline, and monitoring results. In addition, it is important to distinguish between the experimentation pipeline and the production pipeline, the latter including all stages of model creation and not just the final result of the experimentation pipeline. Thanks to the above elements, we can guarantee the iterative nature of MLOps to get the best model.

In relation to ML domain, the issue of task automation requires a comprehensive and robust approach, the concept of autoML [21], which refers to the process of automating various stages of the machine learning lifecycle, such as setting hyperparameters and selecting the best model. autoML can be a valuable tool for reducing the technical complexity linked to the choice of algorithm and hyperparameters. Indeed, autoML enables these parameters to be optimized automatically, which can lead to better performing and more efficient ML models. In addition, the use of autoML can simplify the process of re-training the model when necessary, which can be useful in a dynamic and constantly evolving production environment.

The last block in Figure 1 (Continuous Monitoring) concerns the monitoring phase, which plays an essential role in the smooth operation and ongoing performance of deployed ML models. This phase includes the definition of relevant KPIs (Key Performance Indicators) that measure model performance in real-time. A monitoring dashboard is also set up to visualize these KPIs and provide a clear overview of the status of models in production. The dashboard enables teams to monitor performance, detect anomalies and make decisions based on concrete data. Furthermore, the alert function serves as a notification service, informing the system administrator of critical problems or significant deviations from predefined thresholds. The combination of KPI-based monitoring and an alert function enables proactive intervention, guaranteeing the quality and reliability of deployed models throughout their lifecycle.

2.2 Problem statement

The process of developing ML model doesn't end with the deployment of the model. Indeed, once the model is operational, it is crucial that it continues to deliver quality results and insights over time. So, regular maintenance and updating of the model are key to ensuring its long-term reliability and effectiveness. In addition, continuous monitoring of model performance is also important to detect any problems or changes in the production environment and adapt the model accordingly.

To better understand the challenge of continuous monitoring, an ML model is trained on a specific set of data, but this data may change over time, affecting the model's performance. If a problem is detected with the deployed model, the corrective action to be taken depends mainly on the process involved in setting up an ML project, such as version management of data, models, and hyper-parameters, to be able to re-train a model, for example, when new data is captured.

In this way, monitoring must be cross-functional and detect the essential elements in each phase of the MLOps pipeline. The problem with tools for monitoring ML models lies in the need to select appropriate solutions, to guarantee the reliability, robustness and ongoing performance of these models in real, evolving environments.

3 MONITORING PROCESS IN MLOPS CONTEXT

This section provides a comprehensive overview of monitoring in the context of MLOps, describing the essential steps, metrics and tools to ensure optimal performance and effective maintenance of ML models in production.

In an ML context, the production, delivery and evaluation of a project is a difficult task [7], given its non-deterministic nature, which makes the model complex to test [19], explain and improve over time. This brings us back to the challenge of measuring ML performance indicators that reflect the quality of both the production process and the model itself.

To identify the best ML model for deployment, as illustrated in Figure 2, several series of experiments need to be carried out. It is necessary to define and identify the metrics that can be monitored manually or automatically during the MLOps lifecycle. Furthermore, the monitoring step should be automated to be able to continuously assess the reliability of the models deployed into production.

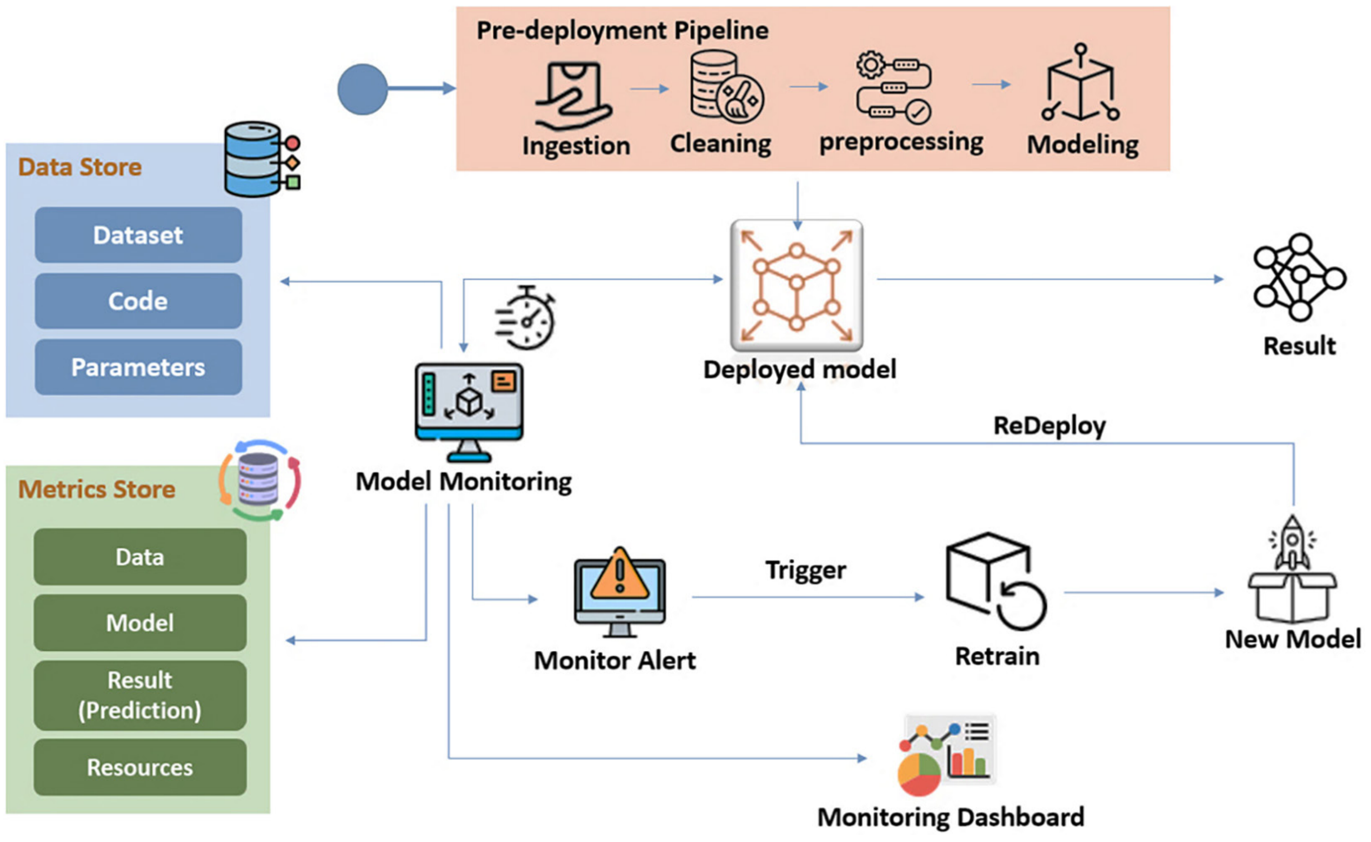


Fig. 2. Process of monitoring in a MLOPS context

Monitoring the workflow in MLOps can identify potential problems, such as models not converging or performance issues in the deployment infrastructure. This enables proactive measures to be taken to improve model quality and reliability, reduce downtime and optimize response time, by triggering the retraining and redeployment of a new model, for example.

4 MONITORING STAGES IN ML PIPELINE

The analogy with an iceberg depicted in Figure 3 highlights the fact that the health of an ML system depends on hidden features that are complex to monitor, such as those related to the data and the model itself, in addition to other more visible elements such as the services provided. Ecosystem health is the most abstract layer, including aspects such as predictive drift [30] and business KPIs, and refers to the overall well-being and functionality of the interconnected elements and processes within a system or organization. The aim is to assess the performance, reliability and stability of this top layer, to ensure that it functions effectively to achieve the desired business results.

Metrics related to monitoring can be classified into gauge metrics, which record the values of the system's gauges, delta metrics, which calculate the variance between previous and current measurements, and cumulative metrics, which track the evolution of counters over time [17].

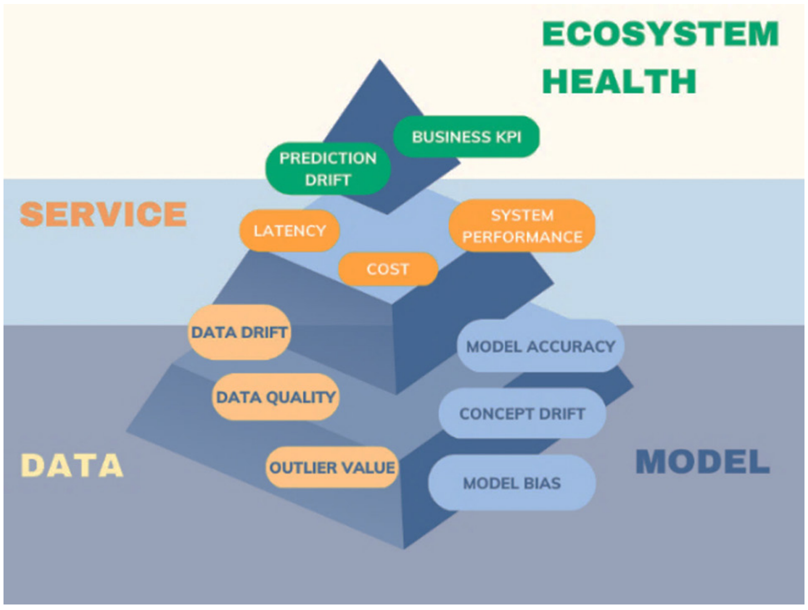


Fig. 3. Hidden and visible features related to monitoring [9] (adapted by the author)

There are several aspects for monitoring an ML model:

- Monitoring model quality and performance in production: evaluate the accuracy of the model's predictions, and see if it declines over time, and if the model needs to be re-trained.
- Monitoring the distribution of model inputs and outputs: check the conformity of the distribution of input data and model hyper-parameters. Observing the distribution of predicted classes over time, which can be linked to data and concept drift, the tool Great Expectations [25] can be used as platforms for verifying data consistency and distribution.

- Monitoring model training and retraining: learning curves, the distribution of trained model predictions or the confusion matrix during training and retraining. The Seldon Alibi-Detect [26] tool can be used, for example, to label abnormal data inputs where model predictions may be unreliable [20].
- Monitoring hardware resources and associated metrics: CPU/GPU load and memory usage metrics are essential for monitoring system performance and efficiency.
- Evaluation of CI/CD pipeline tasks: log metrics, graphs, predictions and other metadata for automated evaluation or test pipelines.

5 MONITORING METRICS

The implementation of monitoring models generally involves the definition and recording of several indicators, each of which specializes in the calculation of a given measure:

The following figure (Figure 4) shows schematically the elements that need to be monitored at each stage of the MLOps pipeline:

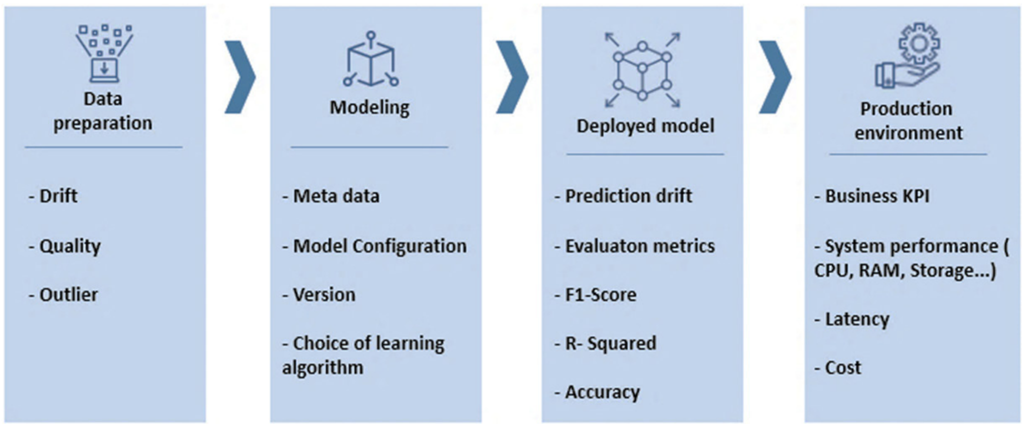


Fig. 4. Elements to monitor by nature of items

MLOPS-related metrics can be spread throughout the development life cycle as well as the production cycle of a ML project. These metrics vary according to need but also according to the stage of the project. For instance, four main stages are of interest: (1) Data preparation, (2) Data Modeling, (3) Deployed model and (5) Production environment. These metrics are explained in the following sections.

5.1 Data preparation concerning the data collection, initial data analysis, data transformation and cleaning and feature engineering

Metrics related to data quality: these metrics are used to assess the quality of the data used to train the model.

Data preparation is the first essential step in creating a successful ML model [13]. This step involves analyzing the data to gain a clear idea of the elements collected and their metadata.

This activity is based on data profiling, which involves checking data quality, such as detecting duplications, inconsistencies, and lack of accuracy, as well as checking attribute values against expected ranges.

According to Felix Naumann's [12], data profiling tasks (Figure 5) can be subdivided into single or multiple data sources, and into single or multiple columns.

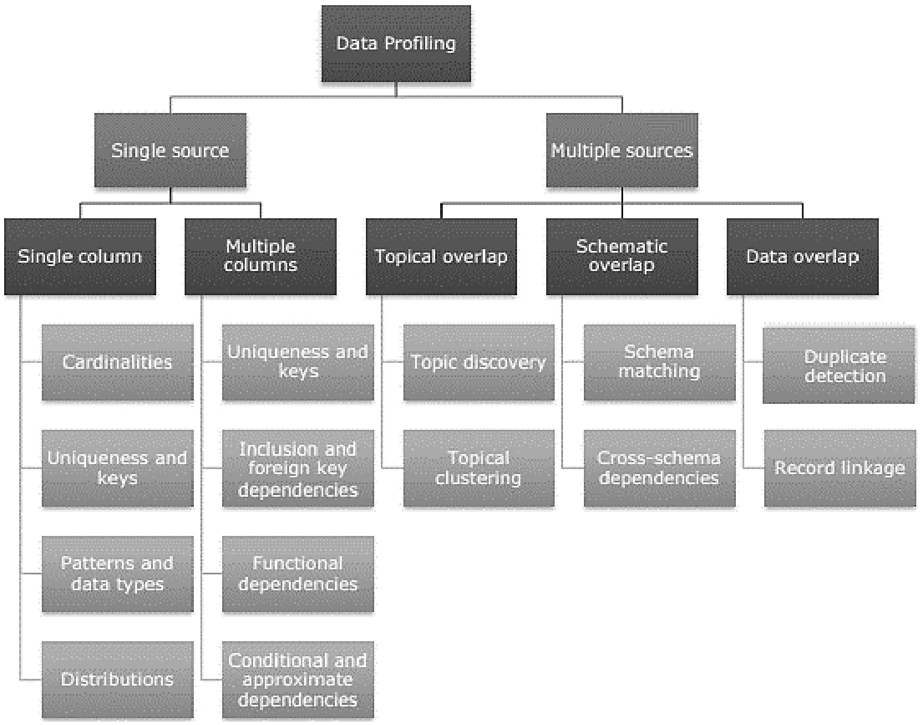
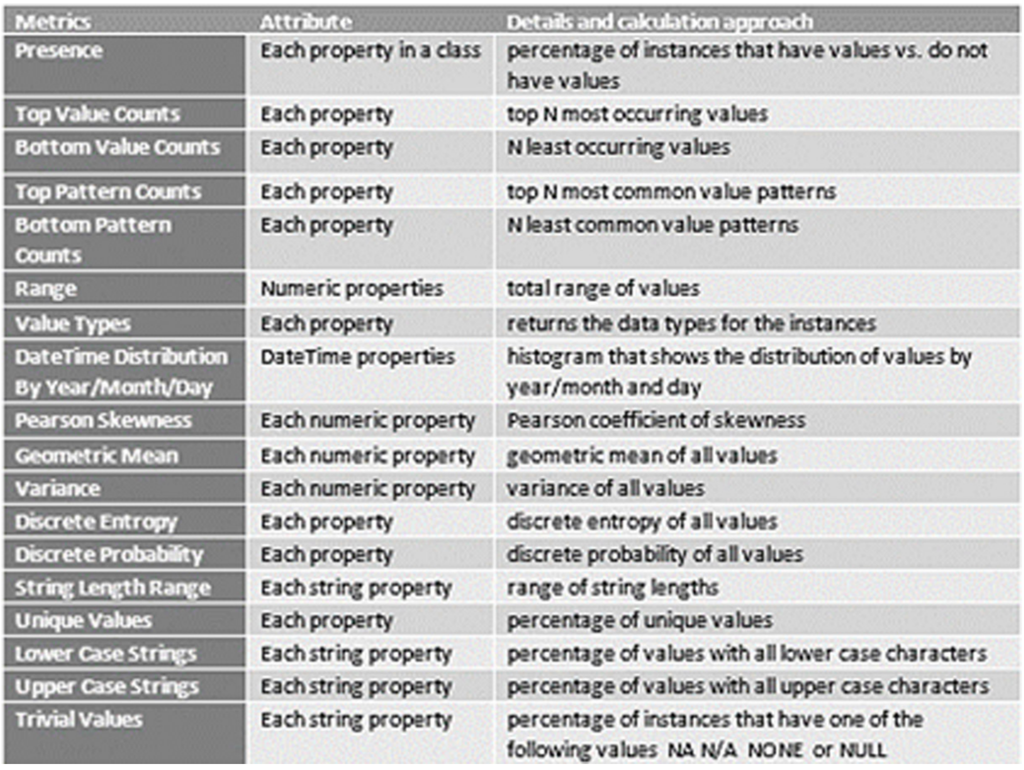


Fig. 5. Classification of data profiling tasks by data nature [12]

During this phase, it is possible to deduce several indicators and metrics that can be used to monitor data quality and make decisions accordingly, such as imputing values and correcting statistical anomalies. The following Table 1 describes the main metrics related to data quality.

Table 1. Data quality/profiling metrics [14]



Metrics	Attribute	Details and calculation approach
Presence	Each property in a class	percentage of instances that have values vs. do not have values
Top Value Counts	Each property	top N most occurring values
Bottom Value Counts	Each property	N least occurring values
Top Pattern Counts	Each property	top N most common value patterns
Bottom Pattern Counts	Each property	N least common value patterns
Range	Numeric properties	total range of values
Value Types	Each property	returns the data types for the instances
DateTime Distribution By Year/Month/Day	DateTime properties	histogram that shows the distribution of values by year/month and day
Pearson Skewness	Each numeric property	Pearson coefficient of skewness
Geometric Mean	Each numeric property	geometric mean of all values
Variance	Each numeric property	variance of all values
Discrete Entropy	Each property	discrete entropy of all values
Discrete Probability	Each property	discrete probability of all values
String Length Range	Each string property	range of string lengths
Unique Values	Each property	percentage of unique values
Lower Case Strings	Each string property	percentage of values with all lower case characters
Upper Case Strings	Each string property	percentage of values with all upper case characters
Trivial Values	Each string property	percentage of instances that have one of the following values NA N/A NONE or NULL

5.2 Modeling and deployed model regarding experimentation and development and also production

During the training, validation and deployment stages of ML model, the metrics most commonly used to evaluate performance, are summarized in this illustration Figure 6. These indicators are used to measure and evaluate the performance of the machine learning model [10]:

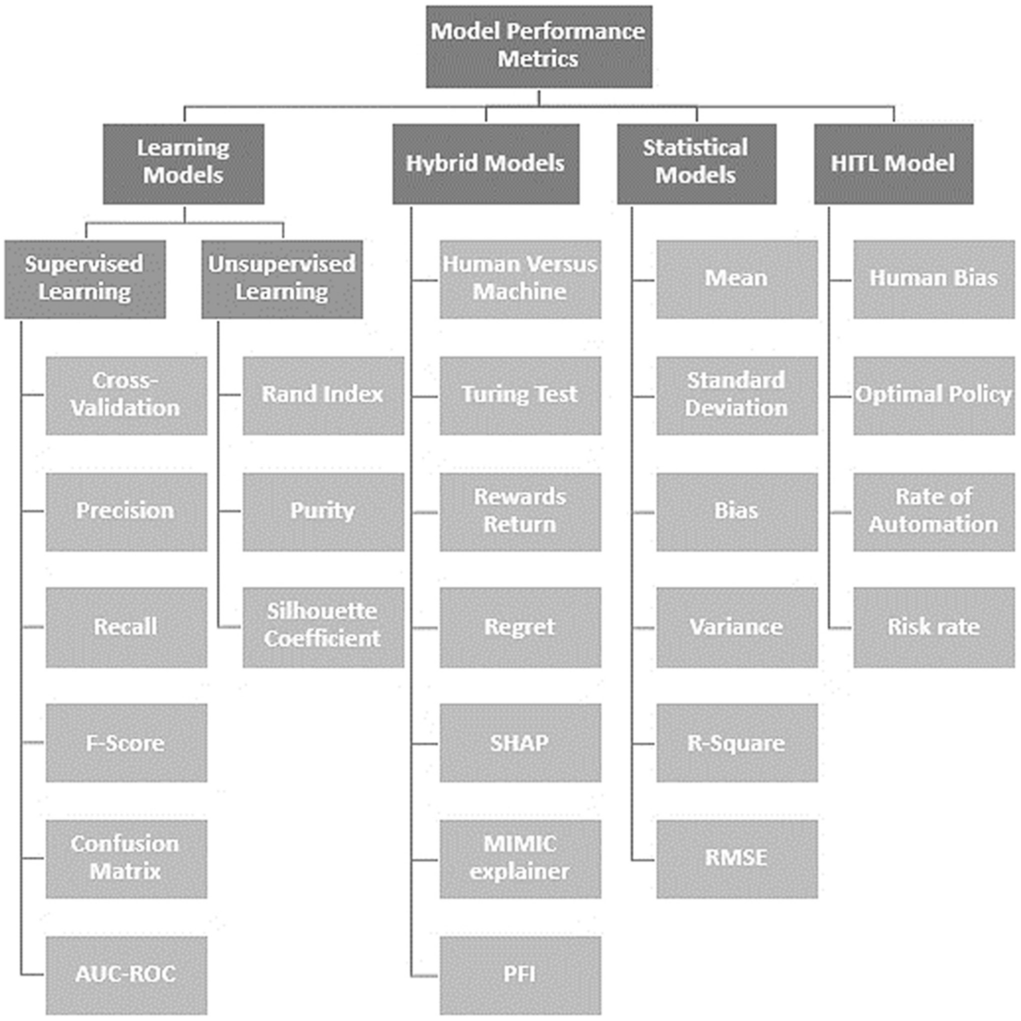
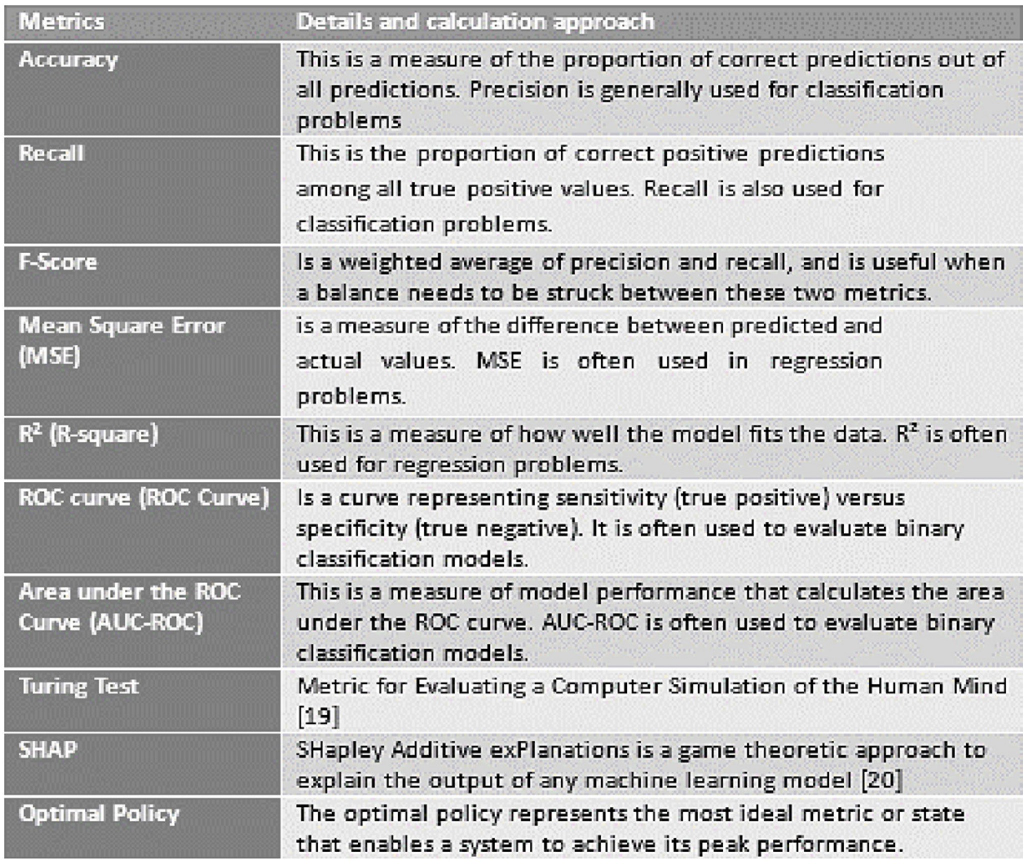


Fig. 6. Evaluation methods and metrics by category and model type [18]

As part of the process of managing MLOps, model evaluation and interpretability are essential components. These fundamental aspects enable us to understand and validate ML models, with a view to determining their added value. Given the plurality of machine learning models, a wide range of evaluation techniques specific to each type of model is available Figure 6.

In the following, an explanation and details of the most commonly used metrics for training ML models (Table 2) are given.

Table 2. Metrics related to training and production



Metrics	Details and calculation approach
Accuracy	This is a measure of the proportion of correct predictions out of all predictions. Precision is generally used for classification problems
Recall	This is the proportion of correct positive predictions among all true positive values. Recall is also used for classification problems.
F-Score	Is a weighted average of precision and recall, and is useful when a balance needs to be struck between these two metrics.
Mean Square Error (MSE)	is a measure of the difference between predicted and actual values. MSE is often used in regression problems.
R^2 (R-square)	This is a measure of how well the model fits the data. R^2 is often used for regression problems.
ROC curve (ROC Curve)	Is a curve representing sensitivity (true positive) versus specificity (true negative). It is often used to evaluate binary classification models.
Area under the ROC Curve (AUC-ROC)	This is a measure of model performance that calculates the area under the ROC curve. AUC-ROC is often used to evaluate binary classification models.
Turing Test	Metric for Evaluating a Computer Simulation of the Human Mind [19]
SHAP	SHapley Additive exPlanations is a game theoretic approach to explain the output of any machine learning model [20]
Optimal Policy	The optimal policy represents the most ideal metric or state that enables a system to achieve its peak performance.

5.3 Production environment or deployment area

Usage and efficiency metrics: These metrics are used to evaluate the use of the model in production. Common metrics include query rate, response time, latency, prediction speed, resource consumption, etc.

User satisfaction metrics: These metrics are used to assess end-user satisfaction. These metrics include satisfaction rate, return on investment ROI, as well as other benefit indicators linked to the context and activity of the company or individuals using the ML model.

Having defined the metrics and indicators for evaluating ML models that are most appropriate to the context in which they are used, it's essential to talk about the right tools for monitoring. These tools enable data to be collected, visualized and analyzed in real-time, offering complete visibility over the performance of the models deployed. By using the right monitoring tools, teams can be proactive in managing ML models, ensuring optimal performance and making informed decisions to maintain quality and reliability throughout the model lifecycle.

6 MONITORING TOOLS

The choice of monitoring tool depends on the features and elements to be monitored. Some features are decisive, such as whether the tool is open source and free of charge, or whether its use generates additional costs. Another essential factor is ease of interconnection with the system and with other training and model deployment tools.

The Table 3 below provides a summary of most established MLOps tools in the field of monitoring ML models.

Table 3. Comparison of ML project monitoring tools [11]

	Neptune.ai	Arize	WhyLabs	Grafana + Prometheus	Evidently	Qualdo	Fiddler	Amazon SageMaker	Seldon Core	Censius
Model evaluation and testing	Yes	Limited	Limited	No	No	No	No	No	No	No
Hardware metrics	Yes	No	No	Yes	No	No	No	Yes	No	No
Model input/output distribution	No	Yes	Yes	Limited	Yes	Yes	Yes	Yes	Yes	Yes
Model training and re-training	Yes	Limited	Limited	No	No	No	No	Yes	No	No
Model performance in production	No	Yes	Yes	Limited	Yes	Yes	Yes	Yes	Yes	Yes
CI/CD pipelines for ML	Yes	No	No	No	No	No	No	Yes	No	No
Open Source	No	No	No	Yes	Yes	No	No	No	No	No

The monitoring tool must integrate as many monitoring functions as possible, such as monitoring data drift, characteristics of data and models, as well as the ability to compare several models running at the same time. A/B testing, for example, stands for comparing two or more model variations to see which performs better. Another essential element is that the tool must provide the ability to program automatic alerts and notifications in the event of a problem being detected. In order to dive into a detailed comparison of ML project monitoring tools, we propose the following radar chart depicting the main characteristics of some open-source and non-open-source monitoring tools for ML models

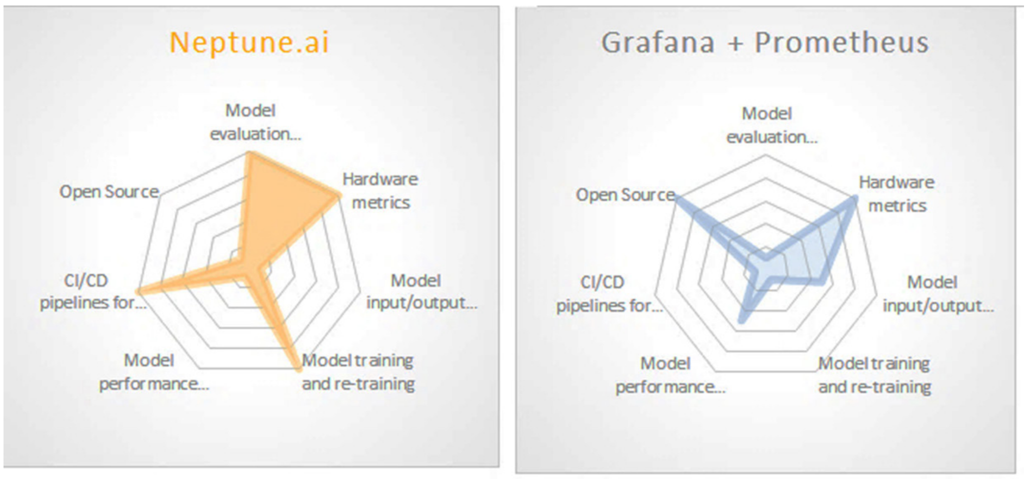


Fig. 7. *(Continued)*

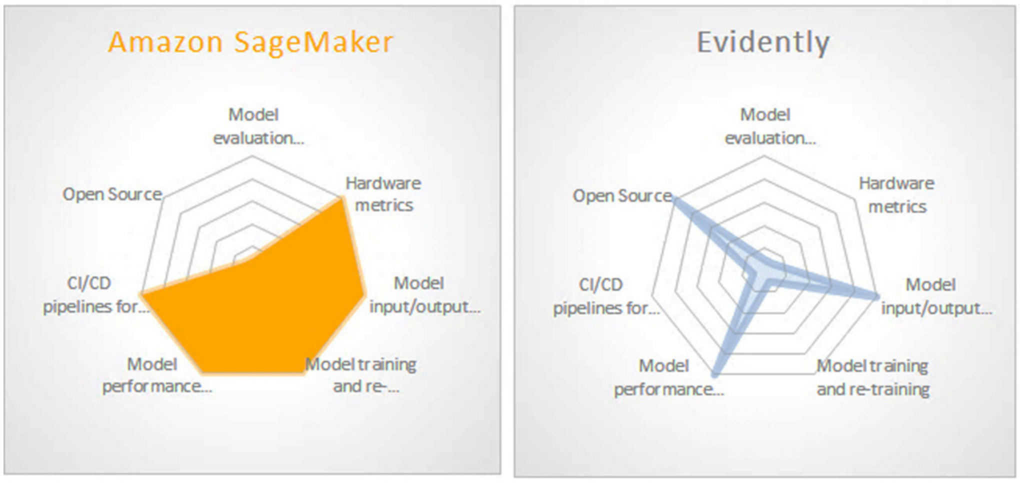


Fig. 7. Radar chart of the main characteristics of some open-source and non-open-source monitoring tools for ML models

Proprietary monitoring tools stand out for their broad coverage in terms of the features mentioned, compared with open-source tools. This is why it is essential to propose an architecture combining several open-source tools to cover the whole matrix of necessary characteristics. An approach that integrates different open-source solutions makes the most of their specific advantages. For example, by using the MLflow tool [3] to manage the lifecycle of ML models, including training and retraining, we can ensure better traceability and reproducibility of results. Combining the GitHub platform for CI/CD pipelines ensures continuous integration and efficient deployment of models. For monitoring hardware metrics and distributing model inputs/outputs, the use of Grafana/Prometheus proves invaluable. By combining these open-source tools, we can talk about optimizing the management of ML models, while benefiting from a tailor-made, adaptable and high-performance solution.

7 DISCUSSION

With the emergence of the ML field, the focus has been on the problem of creating and training the most reliable models possible. Over time, machine learning models are becoming more and more common in the real world now. With the large mass of data generated worldwide, the challenge is to ensure a smooth deployment from development environment to production environment and also to guarantee very good results over time after deployment of the trained model. Monitoring the ML model in the production environment is the key to ensuring that it fulfills its mission.

To this end, various types of indicators and metrics can be measured. These indicators enable us to judge the quality of the model's predictions in relation to actual data. Other metrics, such as response time, resource consumption or error rate, can also be monitored to ensure that the model does not present any performance or stability problems. By regularly monitoring these indicators and metrics, data scientists can quickly detect any problems and take the necessary steps to remedy them.

There are several options for calculating the various metrics associated with a model, including open-source tools such as Grafana and Prometheus, as well as proprietary solutions. These tools can help to assess how well the model is working, as well as to identify any problems and what needs to be done to resolve them. For an effective monitoring of platform, it is essential to support features such as automated alert management and model training, as well as automated model deployment.

Future research should focus on implementing a complete end-to-end architecture for managing ML projects in the MLOps domain in the context of open-source tools. It is also essential to delve deeper into the notion of monitoring, which plays a crucial role in setting up an industrialized ML project. In our increasingly interconnected world, where data exchange is of paramount importance, improved decision-making can generate significant added value.

8 CONCLUSION

As ML systems mature, model monitoring has become an imperative in the MLOps field. Indeed, the implementation of such a framework is essential to guarantee the reliability and resilience of the ML system. Without such monitoring, the ML system risks losing the trust of its users, with potentially fatal consequences. How you think about monitoring depends on the service provided by the Machine Learning model itself. In the future, we'll be taking an in-depth look at the monitoring phase of the MLOps pipeline, highlighting the various techniques and tools available to ensure this important component.

When choosing the monitoring architecture for ML models, there are several important criteria to consider. Easy integration with model training and deployment tools, as well as flexibility to adapt to specific needs, are important. In addition, the architecture should enable the implementation of the majority of metrics associated with machine learning, while ensuring the automation of alerts based on data and model quality. In short, a suitable monitoring architecture must be able to respond effectively to these different aspects to guarantee high-performance, reliable monitoring of machine learning models.

Looking ahead, future research in this area could focus on several key aspects. Firstly, the exploration of more advanced AI-based monitoring techniques, such as AutoML, to serve monitoring to take into account increasingly complex and dynamic machine learning models. Secondly, examining the development of standardized monitoring protocols and frameworks that can be applied across various sectors, taking into account the different contexts in which machine learning models are deployed.

Future research endeavors should prioritize the development of a comprehensive end-to-end architecture for effectively managing ML projects within the MLOps domain, leveraging open-source tools and embracing advanced CI/CD principles for heightened automation. Furthermore, there is a critical need for a more profound exploration of the concept of monitoring, which plays a pivotal role in establishing industrialized ML projects. In our increasingly interconnected world, where data exchange holds paramount importance, enhanced decision-making capabilities, facilitated by advanced CI/CD practices, have the potential to yield substantial added value.

9 REFERENCES

[1] S. Mäkinen, H. Skogström, E. Laaksonen, and T. Mikkonen, "Who needs MLOps: What data scientists seek to accomplish and how can MLOps help?" *ArXiv [cs.SE]*, 2021. https://doi.org/10.1109/WAIN52551.2021.00024

[2] A. Bodor, M. Hnida, and D. Najima, “MLOps: Overview of current state and future directions,” in *Innovations in Smart Cities Applications*, Volume 6, Cham: Springer International Publishing, 2023, pp. 156–165. https://doi.org/10.1007/978-3-031-26852-6_14

[3] S. Alla and S. K. Adari, “Introduction to MLFlow,” in *Beginning MLOps with MLFlow*, Berkeley, CA: Apress, 2021, pp. 125–227. https://doi.org/10.1007/978-1-4842-6549-9_4

[4] R. J. Priyankasingh and H. Y. Vani, “Deployment and serving ML models using kubeflow and KfServing,” in *Lecture Notes in Networks and Systems*, Singapore: Springer Nature Singapore, 2023, pp. 283–293. https://doi.org/10.1007/978-981-19-9228-5_25

[5] “Data version control · DVC,” *Data Version Control · DVC*. [Online]. Available: https://dvc.org/. [Accessed: Sept. 07, 2023].

[6] L. Faubel and K. Schmid, *An Analysis of MLOps Practices*. 2023. https://doi.org/10.25528/166

[7] G. Symeonidis, E. Nerantzis, A. Kazakis, and G. A. Papakostas, “MLOps—Definitions, Tools and Challenges,” in *2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, NV, USA, 2022, pp. 453–460. https://doi.org/10.1109/CCWC54503.2022.9720902

[8] A. John, *Internet of Things (IOT): Systems and Applications*, 2023.

[9] A. Olga, G. Monteiro, G. Leite, and V. Lima, “Continuous monitoring,” Github.io. [Online]. Available: https://mlops-guide.github.io/MLOps/Monitoring/. [Accessed: Sep. 10, 2023].

[10] T. Srivastava, “12 important model evaluation metrics for Machine Learning everyone should know (updated 2023),” *Analytics Vidhya*, 06-Aug-2019. [Online]. Available: https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/. [Accessed: Sept. 07, 2023].

[11] J. Czakon, “Best tools to do ML model monitoring,” Neptune.ai, 13-Sep-2022. [Online]. Available: https://neptune.ai/blog/ml-model-monitoring-best-tools. [Accessed: Sept. 07, 2023].

[12] F. Naumann, “Data profiling revisited,” *SIGMOD Rec.*, vol. 42, no. 4, pp. 40–49, 2014. https://doi.org/10.1145/2590989.2590995

[13] D. Mertz, *Cleaning Data for Effective Data Science: Doing the Other 80% of the Work with Python, R, and Command-Line Tools*. 2021.

[14] “Data profiling metrics,” *Cambridgesemantics.com*. [Online]. Available: https://docs.cambridgesemantics.com/anzo/v5.3/userdoc/metrics.htm. [Accessed: Sept. 10, 2023].

[15] S. Shankar, R. Garcia, J. M. Hellerstein, and A. G. Parameswaran, “Operationalizing Machine Learning: An Interview Study,” *arXiv 2022,* arXiv:2209.09125 2022.

[16] D. Sculley, “Hidden technical debt in machine learning systems,” *NIPS*, pp. 2494–2502, 2015.

[17] C. Huyen, *Designing Machine Learning Systems: An Iterative Process for Production-Ready Applications*. O’Reilly Media, 2022.

[18] E. Raj, *Engineering MLOps: Rapidly Build, Test, and Manage Production-Ready Machine Learning Life Cycles at Scale*. Birmingham, England: Packt Publishing, 2021.

[19] N. Alvarado, S. S. Adams, S. Burbeck, and C. Latta, “Beyond the turing test: Performance metrics for evaluating a computer simulation of the human mind,” in *Proceedings 2nd International Conference on Development and Learning. ICDL 2002*, 2002, pp. 147–152.

[20] S. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” *ArXiv [cs.AI]*, 2017.

[21] Z. Huang, P. Subramaniam, R. C. Fernandez, and E. Wu, “Kitana: Efficient data augmentation search for autoML,” *ArXiv [cs.DB]*, 2023.

[22] M. Testi *et al.*, “MLOps: A taxonomy and a methodology,” *IEEE Access*, vol. 10, pp. 63606–63618, 2022. https://doi.org/10.1109/ACCESS.2022.3181730

[23] M. Borg, R. Jabangwe, S. Åberg, A. Ekblom, L. Hedlund, and A. Lidfeldt, "Test automation with grad-CAM Heatmaps-A future pipe segment in MLOps for vision AI?" in *2021 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW),* Porto de Galinhas, Brazil, 2021, pp. 175–181. https://doi.org/10.1109/ICSTW52544.2021.00039

[24] G. Fursin, "The collective knowledge project: Making ML models more portable and reproducible with open APIs, reusable best practices and MLOps," *ArXiv [cs.LG]*, 2020.

[25] "Data validation workflow," Greatexpectations.io. [Online]. Available: https://docs.greatexpectations.io/docs/guides/validation/validate_data_overview. [Accessed: Sept. 10, 2023].

[26] A. Van Looveren and O. Cobb, "Context-aware drift detection," *Seldon*, 18-Mar-2022. [Online]. Available: https://www.seldon.io/research/context-aware-drift-detection. [Accessed: Sept. 10, 2023].

[27] T. Oladipupo, "Machine learning overview," in *New Advances in Machine Learning*, InTech, 2010. https://doi.org/10.5772/9374

[28] A. H. Aljammal, S. Taamneh, A. Qawasmeh, and H. Bani Salameh, "Machine learning based phishing attacks detection using multiple datasets," *International Journal of Interactive Mobile Technologies,* vol. 17, no. 5, pp. 71–83, 2023. https://doi.org/10.3991/ijim.v17i05.37575

[29] R. H. Jhaveri, A. Revathi, K. Ramana, R. Raut, and R. K. Dhanaraj, "A review on machine learning strategies for real-world engineering applications," *Mobile Information Systems*, vol. 2022, pp. 1–26, 2022. https://doi.org/10.1155/2022/1833507

[30] M. S. Althabiti and M. Abdullah, "CDDM: Concept drift detection model for data stream," *International Journal of Interactive Mobile Technologies,* vol. 14, no. 10, p. 90, 2020. https://doi.org/10.3991/ijim.v14i10.14803

[31] I. Lasri, A. Riadsolh, and M. ElBelkacemi, "Self-attention-based Bi-LSTM model for sentiment analysis on tweets about distance learning in higher education," *International Journal of Emerging Technologies in Learning,* vol. 18, no. 12, pp. 119–141, 2023. https://doi.org/10.3991/ijet.v18i12.38071

10 AUTHORS

Anas Bodor is the Head of the Quality, Standards, and Technological Monitoring Unit at the Moroccan Statistical Office. He is also a PhD student at the LyRica Lab in the School of Information Sciences, Rabat, Morocco (E-mail: anas.bodor@esi.ac.ma).

Meriem Hnida is Assistant Professor at the Information Sciences School, PhD in Computer Sciences at the Mohammadia School of Engineering (2019). Permanent member of the ITQAN research team, LYRICA laboratory, and associate member of "Networking, Modeling and e-Learning (RIME)" research team of Mohammadia School of Engineering. Her research project is about the application of intelligent systems in education, and focuses on the following research areas: Intelligent Tutorial Systems (ITS), Educational Technology, Knowledge Engineering.

Najima Daoudi is a Full Professor at the School of Information Sciences, Rabat, Morocco. She has an engineering degree from the National Institute of Statistics and Applied Economics (INSEA) and a PhD in Computer Science from ENSIAS.