UDC 004.8:004.051:004.738.5
DOI https://doi.org/10.32689/maup.it.2025.1.1

Maryna BAUTINA
Master, Data Scientist, SoftServe
ORCID: *0009-0002-9617-9262*

MONITORING MACHINE LEARNING MODEL DRIFT IN PRODUCTION PIPELINES: METHODS, METRICS, AND DEPLOYMENT CONSIDERATIONS

Abstract. *The relevance of the study is determined by the need to ensure the stability and effectiveness of machine learning models in the context of dynamic changes in data. The problem of model drift – changes in the statistical characteristics of input data or relationships between features and the target variable – leads to a decrease in prediction accuracy. Detecting and monitoring drift in real-time is crucial for maintaining the stability of models, particularly in fields such as finance, healthcare, and cybersecurity, where changes in input data or conditions can significantly affect model performance.*

The aim *of the paper is to investigate methods for monitoring model drift, particularly within the integration of CI/CD pipelines, to ensure their stability in real-world conditions. Special attention is paid to types of drift (data drift, concept drift, label drift) and the metrics used for their detection. The research methods include analyzing existing tools for monitoring and detecting changes in model behavior through the example of financial risk forecasting, as well as evaluating the effectiveness of integrating monitoring into CI/CD.*

The scientific novelty *lies in the proposed comprehensive approach to detecting drift and integrating monitoring into production pipelines using advanced tools such as Google Vertex AI, AWS SageMaker, and TensorFlow Extended, which allow automatic response to changes in data. The use of such technologies improves prediction accuracy and reduces errors in real-world environments. The study confirms the importance of integrating drift monitoring into the continuous process of updating and adapting models to maintain their effectiveness in the context of constantly changing data.*

The conclusions *show that integrating drift monitoring systems into CI/CD pipelines significantly improves the stability and effectiveness of models. Timely detection of drift allows for prompt model adjustments, reducing the likelihood of model degradation. It has been found that for achieving model stability, the automation of monitoring is crucial, as it allows for a prompt response to changes without manual intervention. This enhances the system's efficiency and reduces risks related to the deterioration of prediction quality.*

Key words: *drift monitoring, CI/CD pipelines, MLOps, model stability, automation, TensorFlow Extended, Google Vertex AI, AWS SageMaker, data drift, concept drift, label drift.*

Марина БАУТІНА. МОНІТОРИНГ ДРЕЙФУ МОДЕЛЕЙ МАШИННОГО НАВЧАННЯ В ПРОДУКЦІЙНИХ ПАЙПЛАЙНАХ: МЕТОДИ, МЕТРИКИ ТА АСПЕКТИ РОЗГОРТАННЯ

Анотація. *Актуальність дослідження зумовлена необхідністю забезпечення стабільності та ефективності моделей машинного навчання в умовах динамічних змін даних. Проблема дрейфу моделей – це зміна статистичних характеристик вхідних даних або залежностей між ознаками та цільовою змінною, що призводить до зниження точності прогнозів. Виявлення та моніторинг дрейфу в реальному часі є важливими для стабільної роботи моделей, особливо в таких галузях, як фінанси, охорона здоров'я, кібербезпека, де зміни вхідних даних або умов можуть значно вплинути на результативність моделей.*

Метою статті *є дослідження методів моніторингу дрейфу моделей машинного навчання, зокрема в рамках інтеграції в CI/CD пайплайни для забезпечення їхньої стабільності у реальних умовах. Окрему увагу приділено типам дрейфу (data drift, concept drift, label drift) та метрикам для їх виявлення. Методи дослідження включають аналіз існуючих інструментів для моніторингу та виявлення змін у поведінці моделей на прикладі фінансового прогнозування, а також оцінку ефективності інтеграції моніторингу в CI/CD.*

Наукова новизна *полягає у запропонованому комплексному підході до виявлення дрейфу та інтеграції моніторингу в виробничі пайплайни за допомогою передових інструментів, таких як Google Vertex AI, AWS SageMaker, та TensorFlow Extended, що дозволяє забезпечити автоматичне реагування на зміни в даних. Використання таких технологій підвищує точність прогнозів і зменшує кількість помилок у реальному середовищі. Дослідження підтверджує важливість інтеграції моніторингу дрейфу в безперервний процес оновлення та адаптації моделей для забезпечення їхньої ефективності в умовах постійно змінюваних даних.*

Висновки *показують, що інтеграція систем моніторингу дрейфу в CI/CD пайплайни значно покращує стабільність і ефективність моделей. Своєчасне виявлення дрейфу дозволяє оперативно коригувати моделі, що знижує ймовірність їх деградації. Виявлено, що для досягнення стабільності моделей важлива автоматизація моніторингу, який дозволяє оперативно реагувати на зміни без потреби в ручному втручанні. Це підвищує ефективність системи та знижує ризики, пов'язані з погіршенням якості прогнозів.*

Ключові слова: *моніторинг дрейфу, CI/CD пайплайни, MLOps, стабільність моделей, автоматизація, TensorFlow Extended, Google Vertex AI, AWS SageMaker, data drift, concept drift, label drift.*

Problem statement. In today's context of widespread implementation of machine learning systems in production processes and services, the problem of ensuring the stability and reliability of their operation in a dynamic environment is of particular relevance. One of the critical threats to the quality of predictions of such models is the phenomenon of drift, i.e., a change in the statistical characteristics of input data or dependencies

between features and the target variable, which leads to a gradual decrease in the accuracy and reliability of the model. Drift can be caused by external factors such as changes in user behavior, market fluctuations, technological updates, and internal factors such as the accumulation of errors or imperfect initial training. Ignoring this process can lead to significant losses for businesses that rely on automated solutions, particularly in finance, healthcare, cybersecurity, logistics, and e-commerce.

The problem is the lack of a universal approach to detecting, measuring, and responding to model drift in the production environment. Most existing systems do not provide a built-in mechanism for monitoring model performance after its deployment, which makes it difficult to detect degradation of results early. In this context, the development and systematization of drift monitoring methods, selecting relevant metrics, and considering the infrastructural aspects of implementing such solutions in real-world conditions are of scientific importance. The study's practical significance is stipulated by the need to create adaptive systems that can identify drift and effectively manage the model's life cycle, ensuring its relevance, compliance with new conditions and minimizing business risks.

Analysis of the latest research and publications. The analysis of scientific research confirms that monitoring the drift of machine learning models in product pipelines covers three main areas: drift detection methods, monitoring metrics and architectures, and model deployment and maintenance aspects.

The first area covers developing and applying methods for detecting data and concept drift in machine learning. The work of D. Eastvedt, G. Naterer, and X. Duan demonstrated using a regression model to detect damage in subsea pipelines by monitoring flow changes. This allows for identifying deviations in hydraulic parameters in real-time [7]. S. Shankar and A. Parameswaran emphasize the concept of observability as a prerequisite for effective drift monitoring in ML systems, proposing the integration of internal and external signals of changes in model behavior [19]. J. Zenisek, F. Holzinger, and M. Affenzeller applied machine learning methods to detect concept drift in predictive maintenance tasks, emphasizing the effectiveness of classifiers sensitive to feature distribution changes [24]. N. Jourdan, T. Bayer, T. Biegel, and J. Metternich focus on deep learning architectures for detecting drift resulting from changes in process parameters during production [12]. S. Ackerman, E. Farchi, O. Raz, M. Zalmanovici, and P. Dube analyze the impact of drift and outliers on the long-term performance of models and propose methods to maintain the stability of forecasts [1]. It is advisable to complement this area by creating hybrid algorithms that integrate the detection of both data drift and changes in model concepts and responses.

The second direction concerns the creation of metrics, tools, and architectures for system monitoring of ML models. P. Kourouklidis, D. Kolovos, J. Noppen, and N. Matragkas propose a model-oriented approach to monitoring, including describing models at the meta-level and using specifications to automatically detect deviations in functioning [13]. B. Derakhshan, A. Rezaei Mahdiraji, T. Rabl, and V. Markl present an architecture for continuously deploying ML pipelines with built-in drift-checking steps as part of the CI/CD process [6]. A. Nandan Prasad describes the components of ML systems supervision within the framework of data governance: logging, alerts, and interpretation of performance changes [17]. D. Wani, S. Ackerman, E. Farchi, X. Liu, and H. Chang focus on the tasks of drift detection in log-analytic pipelines using time series and control over input data distributions [21]. P. Yadav, V. Singh, T. Joffre, O. Rigo, C. Arvieu, E. Le Guen, and E. Lacoste demonstrate an example of drift monitoring in a production system for selective laser melting, where inline monitoring methods are applied, and actual parameters are compared with predicted ones [22]. In their monograph, H. Hapke and C. Nelson systematize the construction of ML pipelines with the inclusion of monitoring mechanisms at all stages – from data collection to forecasting [11]. It is advisable to complement this area by unifying drift metrics, developing alert standards, and integrating Explainable AI tools to interpret the detected changes.

The third area covers deploying, maintaining, and adapting models in changing environments. Y. Yang, Y. Li, T. Zhang, Y. Zhou, and H. Zhang describe an approach to early detection of threats in pipeline systems based on a sensor network and ML models, which demonstrates the need to constantly adapt models to new types of signals [23]. B. C. Vadde and V. B. Munagandla study the transformation of DevOps in the context of ML, proposing implementing MLOps practices with a focus on model maintenance, drift testing, and version control [20]. B. Celik and J. Vanschoren present strategies for adapting AutoML to evolutionary changes in data, offering automatic selection of stable configurations [5]. It is advisable to complement this area by developing frameworks that combine technical scaling, automatic drift detection, and verification of compliance with business goals.

The general analysis shows that monitoring model drift in product ML systems requires an interdisciplinary approach that combines algorithmic methods, engineering practices, and management vision. Further research should be aimed at creating adaptive and transparent systems that can respond to environmental changes promptly without losing model accuracy and interpretability.

Despite significant advances in machine learning model drift detection, several unresolved issues exist. One of the main ones is the limitation of traditional metrics for detecting different types of drift, such as data drift, concept drift, and label drift, which limits the accuracy and timeliness of detecting model changes. In addition,

there is a lack of universal approaches to automatic drift monitoring within MLOps, which creates difficulties when integrating such systems into real-world CI/CD pipelines. Also, the issue of integrating drift monitoring into the continuous model updating process remains open, which is necessary for effective, scalable applications in a real environment.

The proposed research aims to develop new methods and tools for more accurate drift detection, real-time model adaptation, and integration into CI/CD pipelines. This will reduce the risks associated with model degradation and ensure their stability in the face of changing data, which is essential for the further development of science and practical applications in many fields such as finance, healthcare, and cybersecurity.

The purpose of the article is to study the possibilities and risks of using artificial intelligence in education from the perspective of data science to determine how to optimize its implementation, taking into account ethical, technological and social aspects.

To achieve this goal, the following tasks have been identified:

1. Analyze methods for detecting the drift of machine learning models, considering the types of changes (data drift, concept drift, label drift) and the effectiveness of using metrics in machine learning pipelines.

2. Explore approaches to automatic drift monitoring in MLOps, including adaptive model management in the production environment.

3. Determine the importance of continuous integration and delivery (CI/CD) for implementing drift monitoring mechanisms in a reliable and scalable environment.

Summary of the main material. In the process of using machine learning models in production environments, the problem of losing forecasting accuracy due to changes in the statistical characteristics of the data or the patterns in the subject area itself is becoming increasingly relevant. Such changes can be caused by gradual transformations in user behavior, changes in market conditions, technical updates, and random external events. In this context, the study of drift types and appropriate detection methods becomes a key aspect of ensuring the stability and reliability of models within the framework of learning pipeline machines. The most common types of drift are data drift (change in the distribution of input features), concept drift (change in the relationship between features and the target variable), and label drift (change in the distribution of the target variable). Various metrics are used to respond to these changes, including statistical distances, classification errors, and indicators of cluster stability or probability reduction in Bayesian models. Understanding the nature of each type of drift and applying the appropriate tools allows you to build an effective monitoring system in the face of dynamic data (Table 1).

In the current context of the practical use of machine learning models, particularly in automated forecasting systems, recommendation services, financial scoring, or fraud protection, each type of drift can have a different impact on model performance. For example, data drift is typical in e-commerce, where user behavior changes depending on seasons or external factors. In contrast, concept drift is more common in financial models due to changing economic conditions or the emergence of new types of transactions. In such cases, a monitoring system integrated into the MLOps infrastructure can use the above metrics to signal a decline in model quality.

Table 1

Characteristics of machine learning model drift types, methods of detecting them, and relevant metrics

Type of drift	The nature of the data change	Main methods of detection	Examples of applied metrics
Data drift	Changing the distribution of input features without changing the mapping	Distribution analysis, histogram comparison	Kullback-Leibler Divergence, PSI, JS Divergence
Conceptual drift	Changing the relationship between the features and the target variable	Sliding windows, model reconstruction, ensembles	Accuracy decay, Page-Hinkley Test, DDM
Label drift	Changing the frequency or composition of classes in the labels	Monitoring the target variable, comparing frequencies	Hellinger Distance, Label Distribution Comparison

Source: compiled by the authors based on [1; 5; 21; 24]

This allows for timely re-training, adaptive updating, or recalculation of hyperparameters. The distribution of roles in drift detection is also important: developers are responsible for the built-in mechanisms, while DevOps specialists provide logging and transfer of signals to the relevant CI/CD processes. Thus, systematic drift detection based on the classification of its types and relevant metrics is the key to stable and controlled operation of models in real-world conditions.

Automatically detecting and monitoring model drift within the MLOps paradigm is a critical component of the modern machine learning model lifecycle. Once a model is deployed into a production environment,

its quality may change due to changes in input data, user behavior, or external context, resulting in reduced accuracy or disrupted decision-making logic. Therefore, MLOps as an engineering and management paradigm focuses not only on automating the deployment and updating of models but also on continuous monitoring of their performance. In this context, automatic drift monitoring involves the implementation of processes for collecting, processing, analyzing, and signaling changes in model behavior. Such processes are implemented with the help of specialized tools that integrate into data pipelines and interact with CI/CD systems, ensuring continuous control and adaptive model management based on defined thresholds. A key element of this process is the ability to respond in a timely manner, including retraining, updating hyperparameters, replacing the model, or correcting data. In practical terms, these functions are realized through the services of such companies as Google (Vertex AI Model Monitoring) [9], AWS (SageMaker Model Monitor) [3], Microsoft (Azure Monitor) [15], as well as through open-source frameworks such as Evidently AI [8], River [18], or Alibi Detect [2] (Table 2).

Table 2

Methods for automatic monitoring of machine learning model drift in MLOps and adaptive model management in the production environment

Platform / tool	Drift monitoring mechanism	Integration into MLOps / CI/CD	Support for adaptive response
Google Vertex AI	Automatic tracking of data/ concept drift	Full integration with Vertex Pipelines and TFX	Notifications via Pub/Sub, automatic retraining
AWS SageMaker model monitor	Baseline-based monitoring of changes in distributions	Integration with SageMaker pipelines and cloudwatch	Events for Lambda or Step Functions
Microsoft azure monitor	Monitoring model performance via ML telemetry	Connection with Azure DevOps and ML lifecycle management	Manual or automated updates
Evidently AI	Web interface and API for drift monitoring	Embeds in any pipeline via Python SDK	Metrics, alerts, integration with Airflow/Kubeflow
Alibi detect / river	Statistical drift detection in streaming data	Suitable for edge or online environments	Real-time support, event flagging

Source: compiled by the authors based on [2; 3; 4; 8; 9; 15; 18]

The systems presented in Table 2 are developed by leading companies such as Google, AWS, Microsoft, and Meta AI and offer a variety of monitoring approaches to improve the reliability and efficiency of models in production environments.

For example, Google Vertex AI provides powerful tools for real-time model monitoring, including data and concept drift detection, and integrates with other Google Cloud components to provide full automation of pipelines [9]. The TensorFlow Extended (TFX) platform allows building scalable pipelines that include automatic detection of data changes, adaptive learning, and model updates, which ensures stability at all stages of the model cycle [10].

Amazon SageMaker Model Monitor is another powerful tool for automatic drift detection that compares current data distributions with baseline profiles to identify deviations, allowing timely model adjustments and preventing model degradation [3]. In particular, AWS Machine Learning Lens offers recommendations for building architectures for ML systems with built-in mechanisms for monitoring and responding to changes [4].

Microsoft Azure offers comprehensive solutions for model lifecycle management through Azure Machine Learning, including continuous monitoring and automatic model updates. The platform supports integration with Azure DevOps, which provides synergy between monitoring and CI/CD processes [15]. Additionally, Microsoft Responsible AI emphasizes the importance of adaptive model management with ethical standards, which helps to ensure the stability and correctness of model behavior in a changing environment [16].

Meta AI, on the other hand, uses Fully Sharded Data Parallel, which allows efficient management of large models and reduces the load on resources while maintaining the stability and accuracy of results even in the face of significant changes in input data [14].

Thus, the approaches presented in Table 2 reflect current practices in the automatic monitoring of model drift in the context of MLOps, allowing them to adapt and optimize models in the face of changing data and external factors, which is important to ensure their stable operation in the production environment.

Integrating drift monitoring is critical in today's continuous integration and delivery (CI/CD) environment for machine learning models. CI/CD mechanisms create an infrastructure that allows you to automatically update models, track their performance, and ensure stability in the face of changing data. This lets you quickly detect deviations from optimal model performance and take the necessary measures to maintain its reliability and efficiency. The practical example of using CI/CD for drift monitoring allows us to realize this idea through

an automatic process that includes constant updating, performance checking, and detection of changes in model behavior, which can significantly improve its stability in a real-world environment.

As part of the experimental study, a practical case of implementing drift monitoring of machine learning models in the financial sector was implemented. The business problem arose after the launch of a new loyalty program: the bank faced a sharp increase in the number of customers who fell into the risk zone, despite high predicted solvency indicators. This led to suspicions that the credit risk forecasting model was degraded. The study collected transactional data for the last 12 months, behavioral metrics (frequency of payments, regularity of spending, average amount), and basic demographic characteristics (age, region, income level). XGBoost's model was built on historical data with a «risky/non-risky customer» target. After its training, it was integrated into the CI/CD pipeline using TensorFlow Extended (TFX), and monitoring was implemented through Google Vertex AI. Table 3 shows a comparison of the tools used in the case study with a description of the monitoring methods and the results of their application.

Table 3

Comparison of tools for monitoring model drift within CI/CD pipelines and results of their use in the case study

Tool description	Methods of drift monitoring	Approach to integration with CI/CD	Results of the experiment
Cloud-based platform for ML models and monitoring	Automatic detection of data and concept drift	Integration via Vertex Pipelines	Changes in the age distribution and behavioral patterns were detected, retraining was initiated
A platform for creating full-fledged ML pipelines	Tracking drift at the preparation and evaluation stages	CI/CD integration via TFX Pipelines	The model was updated to reflect the changes in features, and the history of changes was saved
AWS platform for ML models and distribution monitoring [3].	Comparison with the baseline profile, monitoring	Integration with AWS CodePipeline	Increase in drift errors confirmed, model replaced with a backup model, errors reduced

Source: compiled by the authors based on [3; 6; 9; 10; 17]

During the experiment, the performance of the model was monitored in a real environment for four months. Google Vertex AI [9] revealed a significant data drift: the share of young borrowers (under 30) increased by 18%, which affected the model's accuracy. Concept drift manifested itself in a change in the relationship between income and probability of default: customers with official average income became more likely to default after the launch of the deferred payment program. Label drift also occurred: the share of «risky» cases increased by 12% compared to the training sample [21].

Thanks to the integrated monitoring system, drift alerts were automatically generated, the process of re-training the model on updated data was initiated, and the model version in the production environment was automatically updated. After the update the average error rate in risk classification decreased by 15%.

This case study proved that in the absence of automated drift monitoring, the bank would have continued to make erroneous loan decisions, which could have led to increased financial losses. Instead, the CI/CD infrastructure with built-in monitoring made it possible to quickly adapt the model to new realities and avoid critical degradation of the quality of forecasts.

To better illustrate the process of model degradation and its improvement after automatic retraining, Figure 1 shows the change in the model's error rate over a four-month observation period. The data clearly reflect the growing performance deterioration caused by various types of drift and the subsequent recovery following the implementation of monitoring and CI/CD-based model updating. (fig. 1).

As shown in the chart, the model's classification error increased progressively from 22% to 28% over four months, reflecting a gradual decline in accuracy due to data drift (e.g., demographic shifts), concept drift (changes in feature-target relationships), and label drift. These changes remained undetected without monitoring and could have led to further forecasting errors. However, after automated retraining was triggered through Google Vertex AI and CI/CD pipelines, the error rate dropped to 12%. This confirms the effectiveness of integrated drift monitoring systems in stabilizing model performance and mitigating prediction risks in real-time production environments.

Detecting and interpreting machine learning model drift is accompanied by several challenges that limit the effectiveness of existing approaches. One of the main difficulties is the limitation of traditional metrics for drift detection. Many metrics, such as Kullback-Leibler divergence or Hellinger Distance, cannot accurately

capture model behavior changes under complex data changes [17]. They often focus only on superficial changes in the distribution of features, while changes in concepts or interdependencies between variables may be more critical.

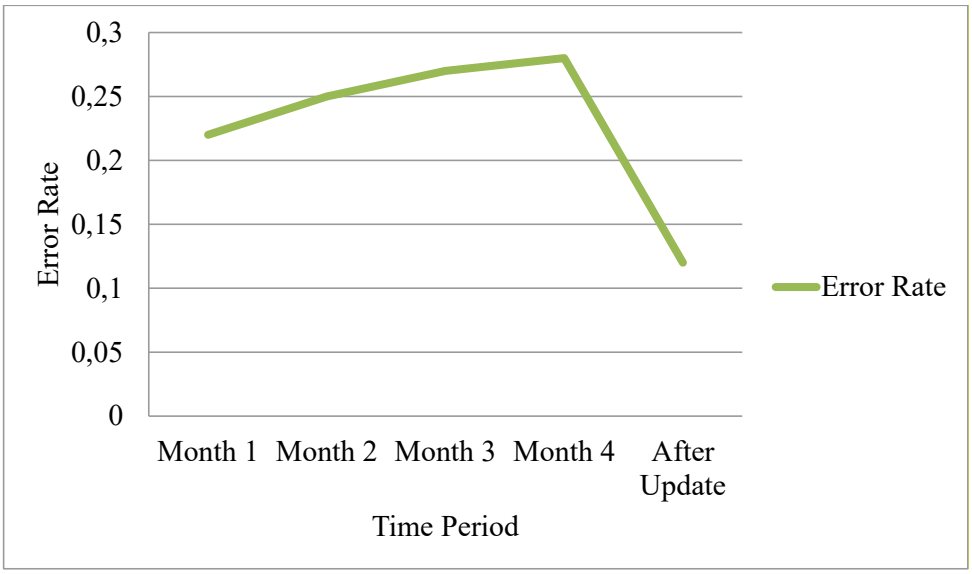

Fig. 1. Model Error Rate Over Time

Source: compiled by the author

Another problem is the difficulty of modeling changes in data, as they can be unpredictable and not always linear. Traditional models often do not consider contextual or cyclical changes, typical in financial or social systems where data may experience periodic fluctuations or spikes. This makes it difficult to develop adaptive algorithms that can instantly respond to these changes [13].

Finally, the absence of a single framework for the practical application of drift monitoring in complex environments changes the approach to integrating such systems into production environments. Existing tools do not always provide easy integration into CI/CD pipelines or have limited support for specific domains. This makes it difficult to create scalable, versatile solutions that can be adapted to different industries or model types.

Integrating a drift monitoring system into the machine learning model development cycle is a prerequisite for ensuring the reliability and stability of models in a real-world environment. This allows the timely detection of data changes and the adaptation of models to new conditions, which is important for their efficiency and accuracy. Within the modern CI/CD paradigm, it is important to automate the monitoring process so that the system can continuously monitor data and model performance without intervention from developers. Integration with CI/CD allows you to automatically update models to correct deficiencies without the need for manual intervention, significantly increasing work efficiency and reliability.

Drift monitoring in CI/CD involves using tools to detect changes in input data distribution or model behavior automatically. Detecting such changes is key to maintaining model stability in changing data. The use of specialized metrics to assess drift, such as changes in forecast accuracy or values of conceptual drift metrics, allows early detection of the need to retrain or optimize the model. An important aspect is the ability to automatically update models using built-in processes, such as retraining or adjusting hyperparameters, without the involvement of human resources, which increases the efficiency of the process.

Integrating monitoring into a single model management system is essential to ensure continuous quality control. CI/CD-enabled platforms allow you to automatically capture change history and set up testing and verification processes. This allows the quality of models to be controlled at all stages of their life cycle and promptly to changes in data or external conditions. In large data volumes and high processing time requirements, the system must scale and process data in real time, reducing delays and ensuring that models are updated promptly when drift is detected.

In general, the integration of drift monitoring into CI/CD pipelines creates the basis for building stable and adaptive machine learning models that can effectively respond to changes in data and maintain high accuracy of predictions in a dynamic environment.

Conclusions. The study confirmed that integrating drift monitoring into CI/CD pipelines is essential for maintaining machine-learning models' stability and accuracy under dynamic data change conditions. Based on a practical case in the banking sector, it was shown that timely detection of data, concept, and label drift enabled a rapid response to changes in customer behavior and preserved the reliability of risk assessment models.

The main challenges identified include the limitations of traditional metrics -such as Kullback-Leibler divergence and Hellinger Distance – which often fail to reflect complex behavioral shifts in data and the lack of universal frameworks for integrating monitoring tools into diverse production environments.

Future research should focus on developing improved monitoring algorithms and flexible integration solutions that support the real-time adaptation of models. Enhancing automated responses to data drift will ensure the long-term reliability of machine-learning systems.

The experimental results demonstrated that applying platforms like Google Vertex AI and TensorFlow Extended reduced classification errors by 15% by enabling automatic retraining and adjusting models in response to drift, highlighting the practical value of continuous monitoring mechanisms.

Bibliography:

1. Ackerman S., Farchi E., Raz O., Zalmanovici M., Dube P. Detection of data drift and outliers affecting machine learning model performance over time. *arXiv preprint*. 2020. arXiv:2012.09258. DOI: https://doi.org/10.48550/arXiv.2012.09258 (date of access: 04.04.2025).

2. Alibi Detect. *Explainable AI for Drift Detection and Model Monitoring*. 2025. URL: https://github.com/SeldonIO/alibi-detect (date of access: 03.04.2025).

3. Amazon Web Services. Amazon SageMaker Model Monitor. 2025. URL: https://docs.aws.amazon.com/sagemaker/latest/dg/model-monitor.html (date of access: 03.04.2025).

4. Amazon Web Services. *Machine Learning Lens – AWS Well-Architected Framework*. 2025. URL: https://docs.aws.amazon.com/wellarchitected/latest/machine-learning-lens/wellarchitected-machine-learning-lens.pdf (date of access: 03.04.2025).

5. Celik B., Vanschoren J. Adaptation Strategies for Automated Machine Learning on Evolving Data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2021. Vol. 43, № 9. P. 3067–3078. DOI: https://doi.org/10.1109/TPAMI.2021.3062900 (date of access: 04.04.2025).

6. Derakhshan B., Rezaei Mahdiraji A., Rabl T., Markl V. Continuous Deployment of Machine Learning Pipelines. *Proceedings of the International Conference on Extending Database Technology (EDBT)*. 2019. P. 397–408. URL: https://hpi.de/fileadmin/user_upload/fachgebiete/rabl/publications/2019/ContinousMLDeploymentEDBT2019.pdf (date of access: 04.04.2025).

7. Eastvedt D., Naterer G., Duan X. Detection of faults in subsea pipelines by flow monitoring with regression supervised machine learning. *Process Safety and Environmental Protection*. 2022. Vol. 161, № 1. P. 409–420. DOI: https://doi.org/10.1016/j.psep.2022.03.049 (date of access: 04.04.2025).

8. Evidently AI. *Monitoring ML Models in Production*. 2025. URL: https://evidentlyai.com (date of access: 03.04.2025).

9. Google Cloud. Vertex AI: Model Monitoring. 2025. URL: https://cloud.google.com/vertex-ai/docs/model-monitoring/using-model-monitoring (date of access: 03.04.2025).

10. Google. *TFX: A TensorFlow-Based Production-Scale Machine Learning Platform*. 2025. URL: https://arxiv.org/abs/1707.01967 (date of access: 03.04.2025).

11. Hapke H., Nelson C. *Building machine learning pipelines*. O'Reilly Media. 2020. 389 p. URL: https://books.google.com.ua/books?id=H6_wDwAAQBAJ (date of access: 04.04.2025).

12. Jourdan N., Bayer T., Biegel T., Metternich J. Handling concept drift in deep learning applications for process monitoring. *Procedia CIRP*. 2023. Vol. 120, № 1. P. 33–38. DOI: https://doi.org/10.1016/j.procir.2023.08.007 (date of access: 04.04.2025).

13. Kourouklidis P., Kolovos D., Noppen J., Matragkas N. A Model-Driven Engineering Approach for Monitoring Machine Learning Models. *MODELS-C: ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion*. 2021. P. 160–164. DOI: https://doi.org/10.1109/MODELS-C53483.2021.00028 (date of access: 04.04.2025).

14. Meta AI. Fully Sharded Data Parallel: faster AI training with less GPU memory. 2025. URL: https://engineering.fb.com/2021/07/15/open-source/fsdp/ (date of access: 03.04.2025).

15. Microsoft Azure. Model management and deployment concepts. 2025. URL: https://learn.microsoft.com/en-us/azure/machine-learning/concept-model-management-and-deployment (date of access: 03.04.2025).

16. Microsoft. *Responsible AI Standard – White Paper*. 2025. URL: https://www.microsoft.com/en-us/ai/responsible-ai?activetab=pivot:techresources-tab (date of access: 03.04.2025).

17. Prasad A. N. Monitoring and Maintaining Machine Learning Systems. In: *Introduction to Data Governance for Machine Learning Systems*. Apress, Berkeley, CA. 2024. P. 129–147. DOI: https://doi.org/10.1007/979-8-8688-1023-7_7 (date of access: 04.04.2025).

18. River. *Online Machine Learning in Python*. 2025. URL: https://riverml.xyz (date of access: 03.04.2025).

19. Shankar S., Parameswaran A. Towards observability for production machine learning pipelines. *arXiv preprint*. 2021. arXiv:2108.13557. DOI: https://doi.org/10.48550/arXiv.2108.13557 (date of access: 04.04.2025).

20. Vadde B. C., Munagandla V. B. DevOps in the Age of Machine Learning: Bridging the Gap Between Development and Data Science. *International Journal of Machine Learning Research in Cybersecurity and Artificial Intelligence*. 2024. Vol. 15, № 1. P. 530–544. URL: https://www.researchgate.net/publication/388526732 (date of access: 04.04.2025).

21. Wani D., Ackerman S., Farchi E., Liu X., Chang H. Data Drift Monitoring for Log Anomaly Detection Pipelines. *arXiv preprint*. 2023. arXiv:2310.14893. DOI: https://doi.org/10.48550/arXiv.2310.14893 (date of access: 04.04.2025).

22. Yadav P., Singh V. K., Joffre T., Rigo O., Arvieu C., Le Guen E., Lacoste E. Inline drift detection using monitoring systems and machine learning in selective laser melting. *Advanced Engineering Materials*. 2020. Vol. 22, № 12. Article 2000660. DOI: https://doi.org/10.1002/adem.202000660 (date of access: 04.04.2025).

23. Yang Y., Li Y., Zhang T., Zhou Y., Zhang H. Early safety warnings for long-distance pipelines: A distributed optical fiber sensor machine learning approach. *Proceedings of the AAAI Conference on Artificial Intelligence*. 2021. Vol. 35, № 17. P. 14991–14999. DOI: https://doi.org/10.1609/aaai.v35i17.17759 (date of access: 04.04.2025).

24. Zenisek J., Holzinger F., Affenzeller M. Machine learning based concept drift detection for predictive maintenance. *Computers & Industrial Engineering*. 2019. Vol. 137, № 1. Article 106031. DOI: https://doi.org/10.1016/j.cie.2019.106031 (date of access: 04.04.2025).