International Journal of Computer Trends and Technology
ISSN: 2231 – 2803 / https://doi.org/10.14445/22312803/IJCTT-V70I9P105

Volume 70 Issue 9, 38-42, September 2022

Original Article

Monitoring the Performance of Machine Learning Models in Production

Satyanarayan Raju Vadapalli

Sr. Engineering Manager, San Francisco Bay Area, California, USA.

Received: 09 August 2022 Revised: 15 September 2022 Accepted: 27 September 2022 Published: 10 October 2022

Abstract *- Machine learning (ML) models have become vital decision-making components for many businesses in the last decade. However, the performance of ML models degrades over time due to multiple economic or environmental factors that can lead to non-optimal decision-making. With organizations having tens or even hundreds of ML models deployed in production, it is important to ensure the models perform the way they were trained to perform. Additionally, models need to be retrained every few weeks or months to adapt to the evolving environment that affects the model performance. In this article, we discuss an approach that can be used to proactively identify issues with model output and inform the developers and data scientists when it's time to retrain the model. Given the importance of input data quality in model performance, our approach's significant attention is coming up with ways to identify data quality issues and take proactive measures to mitigate the associated risks. This monitoring approach is currently deployed on multiple production models, generating automated alerts on models or data drifts, enabling the data scientists to take corrective actions.*

Keywords *- Drift detection, Machine learning, MLOPs, Monitoring, Observability.*

1. Introduction

With the growing popularity of Machine learning operations or ML Ops, deploying and maintaining machine learning models has become streamlined and allows data scientists to focus on rapid model development. To have an end-to-end ML pipeline that is reliable and efficient, teams adopt a set of practices at the intersection of Data Engineering, DevOps, and Machine learning. The ML pipeline consists of collecting and cleansing data, building feature stores, training and validating the model, releasing the model, and monitoring and retraining. ML Ops's responsibility is to ensure it provides an infrastructure that is reliable and can scale as the needs of an organization grow. Each step in the pipeline is important and acts as a building block to create such a robust infrastructure [9].

This paper focuses on ML model performance monitoring in production, one of the most neglected areas. Often teams take a reactive approach, where they do not think of model performance until they discover unfavorable business outcomes because the model is not doing what it is supposed to. Causes could be numerous – the data fed to the model in production could have evolved and been significantly different from what it was during model training, one or more features used for training might no longer be available due to a business process change, irregularities in data due to application bugs, seasonality in business or changes in the economic environment due to recession or pandemic could impact the input data. Users could make an unannounced schema change, the data could be missing at the source, and there could be several similar issues.

Whatever the reason, clearly defined model monitoring processes and tools will ensure the irregularities are proactively caught and addressed. The checks for such cases might include range compliance, data distribution, feature statistics, correlations, data and target drift. Almost every ML model has this inconvenient problem of degrading over time. Data drift into the picture when the model receives data it hasn't seen in training [2]. For example, users coming from different age groups, marketing channels, or geographic locations.

If the real-world patterns change, the concept drift comes into the picture. For example, a global pandemic affecting all customer behavior or a new competing product on the market offering a generous free subscription. It changes how users respond to marketing campaigns. Degradation of model quality is the symptom of both drifts kicking in. We can monitor if the properties of the input data or target function have changed. For example, we can track the distributions for the key model features and the model prediction. Then, trigger an alert if they significantly differ from a past timeframe [3].

Several techniques can be adapted to measure key model performance metrics, including model drifts and data quality drifts. More importantly, it is important to look at the larger aspect of ML observability that allows teams to dig deeper and truly understand how the model works and the cause behind its actions to troubleshoot and enhance its performance. The most basic strategy to know if a predictive model works well is to compare the results against the actual

outcomes [3]. The performance metrics that we choose to monitor should fit the use case. Thus, it's important to track not just the absolute values but also the error distribution. It is also critical to distinguish between occasional outliers and real decay. So, picking the right metrics goes a long way [4].

2. Related Work

This section will review related work in ML monitoring, MLOps and drift detection. Georgios Symeonidis et al. [1] provide general definitions, tools and challenges in MLOps. Tommi Mikkonen et al. [20] explain the steps necessary to train and deploy ML systems to define the importance of MLOps. In terms of the challenges in the data lifecycle, Neoklis Polyzotis et al. [7] summarize some of the interesting research challenges that they encountered and survey the relevant literature from data management and machine learning communities.

There are plenty of papers around drift detection in monitoring for MLOps such as X. Li et al. [10] present a method for detection of changes in the probability distribution of examples, Rosana Noronha Gemaque et al. [12] present a comprehensive overview of approaches that tackle concept drift in classification problems in an unsupervised manner and also include a proposed taxonomy of state-of-the-art approaches for concept drift detection based on unsupervised strategies, Maayan Harel et al. [11] devise a procedure for detecting concept drifts in data-streams that relies on analyzing the empirical loss of learning algorithms, Samuel Ackerman et al. [13] propose a method that utilizes feature space rules, called data slices, for drift detection, Jan Zenisek et al. [14] present a method to detect concept drift in data streams as potential indication for defective system behavior and depict initial tests on synthetic data sets, João Gama et al. [15] study the problem of learning when the distribution that generate the examples changes over time and Anton Dries et al. [16] show how uniform convergence bounds in learning theory can be adjusted for adaptive concept drift detection.

As for the review of the ML model monitoring task, [8] and [9] provide a good guide for anyone new to the field and looking for a series of steps that define these tasks. Similarly, Konstantin Kutzkov et al. [18] discuss how to approach model monitoring to get the most value out of it. Aparna Dhinakaran et al. [17] cover several challenges connected to the availability of ground truth and discuss the performance metrics available to measure models in each scenario. Juhi Ramzai et al. [5] enlighten us on the PSI and CSI, i.e., the Population Stability Index and Characteristic Stability Index, which are among the most important monitoring strategies used in many domains, especially the credit risk domain. One of the Microsoft Azure blogs [26] teaches us to monitor data drift and set alerts when drift is high using a synthetic dataset. Finally, [27] touches upon Variance Inflation Factor (VIF) for collinearity, another important metric that could be used for model monitoring that hasn't been discussed in this paper as a potential solution.

3. Methodology

Organizations suffer from bad data quality; most are aware of the problem but do little to address it. Data cannot be considered high-quality unless fit for decision-making and planning. Almost every industry today claims to be data-driven. Dependence on data and insights to run successful businesses has grown many folds in the last decade or so. Poor data quality is annoying but not something that affects companies in a real sense. However, most people do not see bad data as the root cause of many problems or wrong decisions. Poor data quality affects ML models immensely; hence, it is important to check data quality [8].

The following section describes a two-tiered approach to ML model monitoring:

3.1. Tier one

Focuses on data quality, monitoring the input data for 3 basic metrics.

3.1.1. Volume

A trend analysis of the population over time, either by day or month. Depending on the model, the trend can be monitored within certain buckets to ensure the population mix doesn’t change significantly to affect the model performance.

3.1.2. Averages

Compare the mean values of input variables in production with the mean values of those in training. It again ensures the variables do not drift significantly to affect the model scores.

3.1.3. Missing data

Monitor the input data for missing or null values. Depending on the weight of the input variable, a missing variable could drastically change a model's behavior.

3.2. Tier two

To make sure a Machine learning model is still relevant once it is deployed in production, regular monitoring is important [28]. For a predictive machine learning model to work well in production, the basic assumption is the data in production will be in line with data used during development and validation. Two of the most widely used metrics to study shifts in population distribution are Population Stability Index (PSI) and the Characteristic Stability Index (CSI) [5]. For models trained offline and deployed in production, it is important to know when the model might need retraining. Studying the population distribution drifts consistently can provide a fair idea if the model predictions are still accurate and save organizations from making wrong decisions.

3.2.1. Population Stability Index (PSI)

It is a statistical metric to measure how much a variable has shifted in distribution between two samples or over time. It is widely used for monitoring changes in a population's characteristics and diagnosing possible problems in model performance. PSI was originally developed in risk scorecards for monitoring the changes in the score distribution between an out-of-time validation sample and a modeling sample [6]. PSI can be used in two different ways within the realm of model monitoring.

Feature selection

When developing a machine learning model, PSI can be used to select the right features. If a feature has a lot of variabilities and is prone to rapid changes in distribution, it might not be the right feature for the model as it would need frequent retraining. PSI can be used to proactively determine if a feature is volatile by analyzing the data over a period [6].

Model retraining

It is possible for the model features to evolve and change over time due to many external influences such as economy, geopolitics etc. Though features seem to have reasonable standard deviations during training, that could change when the model is deployed in production for some time. As this happens, the model predictions become less accurate with the current population. PSI can be used to monitor the drift in features and help as an automatic trigger to retrain the model [6].

In the context of Machine learning models, an independent variable is a cause. Its value is independent of other variables, but drift in an independent variable causes drifts in the dependent variable. At the same time, a dependent variable is an effect. Its value depends on the changes in the independent variable. PSI helps us study the drift's independent variables, such as a model score. CSI helps us monitor the independent variables, such as the features used to train the model and inferencing.

The steps involved in calculating PSI:

$$PSI = \sum((\text{Actual\%} - \text{Expected\%}) \times \ln(\text{Actual\%}/\text{Expected\%}))$$

- Sort the scores in a descending order
- Divide the scores into 10 or 20 buckets, also known as deciles.
- Map the frequency of the samples from training and scoring into each bucket.
- Calculate the percentage in each decile for scoring and development population.
- Calculate the difference in % between scoring and training.
- Compute the natural log of scoring % over training %.
- Use the above formulae to calculate the Index for each bucket and then sum the indexes.

Table 1. Population Distribution

deciles	score	frequency scoring	frequency training	scoring% (A)	training% (B)	(A-B)	In(A/B)	PSI
1	< 0.1	61487	369335	65.63%	61.53%	4.10%	0.06	0.003
2	0.10 0.15	20725	181664	22.12%	30.26%	-8.14%	-0.31	0.026
3	0.16 0.20	7112	41767	7.59%	6.96%	0.63%	0.09	0.001
4	0.21 0.25	3164	4295	3.38%	0.72%	2.66%	1.55	0.041
5	0.26 0.30	719	1766	0.77%	0.29%	0.47%	0.96	0.005
6	0.31 0.35	373	898	0.40%	0.15%	0.25%	0.98	0.002
7	0.36 0.40	76	373	0.08%	0.06%	0.02%	0.27	0.000
8	0.41 0.45	22	106	0.02%	0.02%	0.01%	0.29	0.000
9	0.46 0.50	7	58	0.01%	0.01%	0.00%	-0.26	0.000
10	> 0.50	3	28	0.00%	0.00%	0.00%	-0.38	0.000
Total		**93688**	**600290**					**0.077**

Interpretation of results:

- PSI < 0.1: Very slight to no change. No action is required, and we can continue using the existing model.
- 0.1 <= PSI <= 0.2: Slight change. Monitor more frequently to catch any further drifts.
- PSI > 0.2: Significant change. Should not use this model in production anymore. It should be recalibrated/redeveloped.

3.2.2. Characteristic Stability Index (CSI)

Compares the distribution of an independent variable in the scoring sample to that of a development or training sample. CSI helps determine drifts in the distribution of input variables used for scoring over time. The idea is to identify the feature or the input variable that is the cause of drifts in the score distribution. If PSI scores indicate an index score of over 0.2, the next step is calculating the CSI for each input variable using the same method as PSI [5]. Identifying the variables to attribute the shift in overall score helps recalibrate the model or remove the variable altogether.

4. Automating ML monitoring

Monitoring is challenging and is often not the top priority for teams. Automating the monitoring process helps receive alerts before the actual model performance degradation [21]. The following architecture represents 3 broad areas of the Machine learning model lifecycle - development, inference, and monitoring/alerting [24]. This is an implementation aimed at storing scores, analysing drifts, and sending out alerts when the drifts are beyond a threshold.

4.1. Training, validation, and deployment

In the process of developing a machine learning model, the input data is divided into three datasets - training, validation and testing. The model is initially trained on the training dataset to fit the model's parameters. Then the trained model is used to predict using the validation dataset. The goal here is to provide an unbiased data population so the model's hyperparameters can be tuned. Finally, the model is used on a hold-out or test data set, which is never used during training or validation, to provide an unbiased measure of the model's accuracy [22].

Using training and validation data, PSI and CSI are then used to create benchmark distribution statistics for input variables and scores. These benchmarks will be used to compare the distribution in production and generate alerts on drifts. The model is then deployed in production for inferencing.

4.2. Scoring in production

The input variables are used to score in production, and the scores are stored in a database. The statistical models (PSI and CSI) are run on this data periodically to generate the distribution and are compared to the benchmarks generated in the previous section.

4.3. Monitoring and alerting

Thresholds are defined for each input variable and the scores; developers are alerted when any of these cross the threshold. Choosing the right frequency for comparing the distributions and alerting is determined by what fits best for the given model [23].

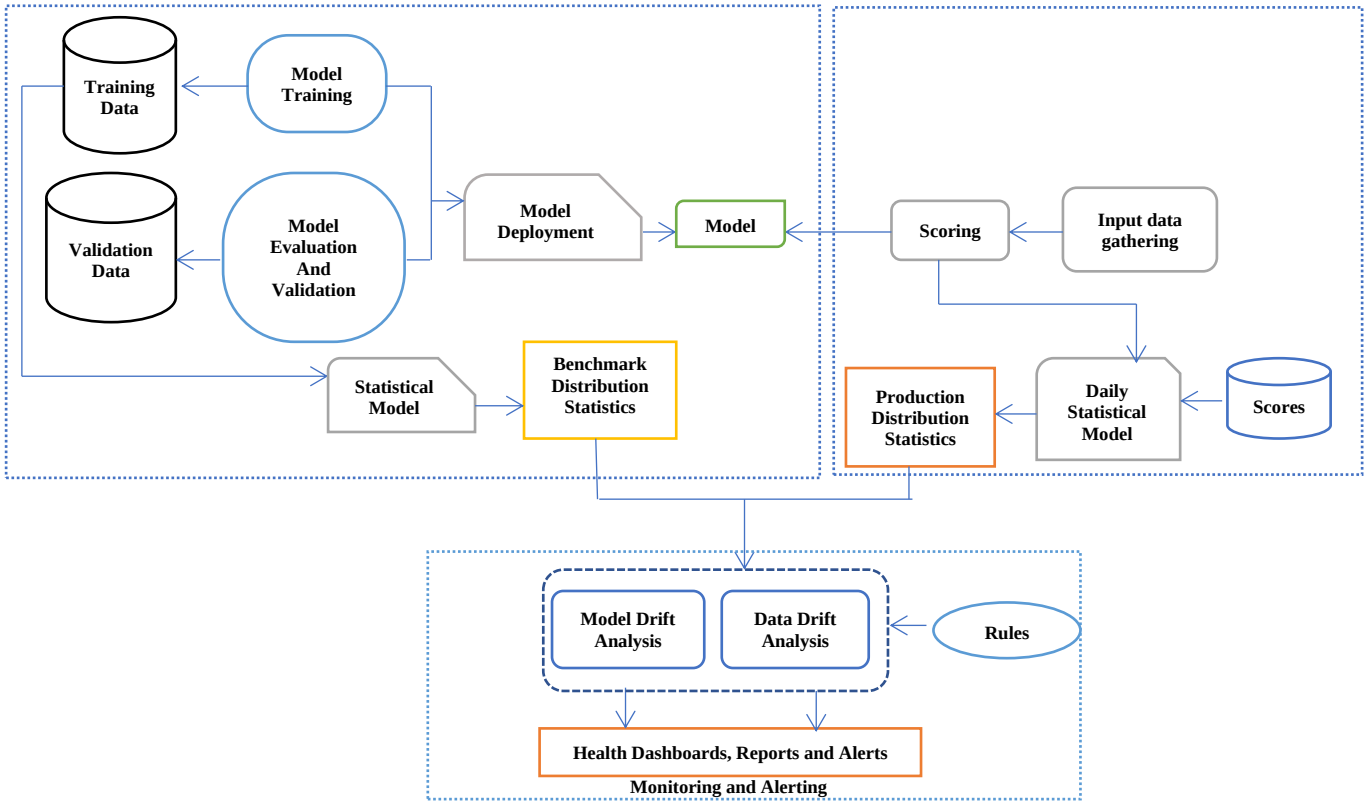

Fig. 1 System Architecture

5. Conclusion

Model monitoring is more than just catching drifts in the distribution. Drifts can happen for various reasons; however, the stakeholders need to detect them and get to the root cause that led to the drift. Proactive determination enables teams to take corrective measures such as retraining the model, resolving any data quality issues upstream or selecting the right features for the model. An efficient monitoring and alerting system can save organizations from making bad decisions based on a suboptimal ML model.

References

[1] Georgios Symeonidis, Evangelos Nerantzis, Apostolos Kazakis, and George A. Papakostas, "MLOps - Definitions, Tools and Challenges," *in Proc. IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC*), pp. 0453-0460, 2022.

[2] Sasu Makinen, Henrik Skogstrom, Eero Laaksonen, and Tommi Mikkonen, "Who Needs MLOps: What Data Scientists Seek to Accomplish and How Can MLOps Help?," *IEEE/ACM 1st Workshop on AI Engineering - Software Engineering for AI (WAIN),* pp. 109-112, 2021.

[3] KDnuggets home, 2021. [Online]. Available: https://www.kdnuggets.com/2021/03/machine-learning-model-monitoring-checklist.html

[4] Altexsoft, 2022. [Online]. Available: https://www.altexsoft.com/blog/machine-learning-metrics/

[5] Towards Data Science, 2021. [Online]. Available: https://towardsdatascience.com/psi-and-csi-top-2-model-monitoring-metrics-924a2540bed8

[6] Alec Zhixiao Lin, *"*Examining Distributional Shifts by Using Population Stability Index (PSI) for Model Validation and Diagnosis*," Western Users of Sas Software(WUSS)*, 2017.

[7] Neoklis Polyzotis, Sudip Roy, Steven Euijong Whang, and Martin Zinkevich, "Data Lifecycle Challenges in Production Machine Learning: A Survey," *ACM SIGMOD Record*, vol. 47, no.2, pp. 17–28, 2018.

[8] Aporia, 2022. [Online]. Available: https://www.aporia.com/machine-learning-model-monitoring-101/

[9] Deepchecks, 2022. [Online]. Available: https://deepchecks.com/how-to-monitor-ml-models-in-production/

[10] X. Li, O.R. Zaiane, and Z. Li (Eds.), "Learning with Local Drift Detection," *ADMA, Lecture Notes in Computer Science*, vol. 4093, pp. 42–55, 2006.

[11] Maayan Harel, Shie Mannor, Ran El-Yaniv, Koby Crammer, "Concept Drift Detection through Resampling," *Proceedings of the 31st International Conference on Machine Learning*, vol. 32, no. 2, pp. 1009-1017, 2014

[12] Rosana Noronha Gemaque, Albert França Josuá Costa, Rafael Giusti, and Eulanda Miranda dos Santos, "An Overview of Unsupervised Drift Detection Methods," *WIREs Data Mining Knowledge Discovery Wiley Periodicals LLC*., vol. 10, no. 6, pp. e1381, 2020.

[13] Samuel Ackerman, Parijat Dube, Eitan Farchi, Orna Raz, and Marcel Zalmanovici, "Machine Learning Model Drift Detection Via Weak Data Slices," *DeepTest workshop of ICSE*, arXiv:2108.05319 [cs.LG], 2021

[14] Jan Zenisek, Florian Holzinger, and Michael Affenzeller, *"*Machine Learning Based Concept Drift Detection for Predictive Maintenance", *Computers & Industrial Engineering*, vol. 137, 2019.

[15] João Gama, Pedro Medas, Gladys Castillo, and Pedro Rodrigues, "Learning with Drift Detection," *Advances in Artificial Intelligence – Brazilian Symposium on Artificial Intelligence (SBIA)*, vol. 3171, pp. 286–295, 2004.

[16] Anton Dries, and Ulrich Rückert, "Adaptive Concept Drift Detection," *Statistical Analysis and Data Mining: The ASA Data Science Journal Wiley*, vol. 2, no. 5-6, pp. 311-327, 2009.

[17] Towards Data Science, 2021. [Online]. Available: https://towardsdatascience.com/the-playbook-to-monitor-your-models-performance-in-production-ec06c1cc3245

[18] Neptune AI, 2022. [Online]. Available: https://neptune.ai/blog/ml-model-performance-monitoring

[19] B. Balaji, Kanagaraj. U, Mahendran. R, Rethinasiranjeevi. R, "Fault Prediction of Induction Motor using Machine Learning Algorithm," *SSRG International Journal of Electrical and Electronics Engineering*, vol. 8, no. 11, pp. 1-6, 2021. Crossref, https://doi.org/10.14445/23488379/IJEEE-V8I11P101

[20] Tommi Mikkonen, Jukka Nurminen, Mikko Raatikainen, Ilenia Fronza, Niko Makitalo and Tomi M, "Is Machine Learning Software Just Software: A Maintainability View In Software Quality Days," *Springer*, vol. 404, pp. 94-105, 2021.

[21] Towards Data Science, 2020. [Online]. Available: https://towardsdatascience.com/how-to-detect-model-drift-in-mlops-monitoring-7a039c22eaf9

[22] Analytics Vidhya, 2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/10/mlops-and-the-importance-of-data-drift-detection/

[23] Deepchecks, 2022. [Online]. Available: https://deepchecks.com/how-to-detect-concept-drift-with-machine-learning-monitoring/

[24] Databricks, 2019. [Online]. Available: https://www.databricks.com/blog/2019/09/18/productionizing-machine-learning-from-deployment-to-drift-detection.html

[25] K. Jino Abisha, J.Roshan Nilofer, A.Silviya, Dr. S. Raja Ratna, "Detection of Twitter Spam's using Machine Learning Algorithm," *SSRG International Journal of Computer Science and Engineering*, vol. 6, no. 3, pp. 10-13, 2019. Crossref, https://doi.org/10.14445/23488387/IJCSE-V6I3P103

[26] Microsoft, 2022. [Online]. Available: https://learn.microsoft.com/en-us/azure/machine-learning/v1/how-to-monitor-datasets?tabs=python

[27] Github, 2017. [Online]. Available: https://etav.github.io/python/vif_factor_python.html

[28] Sigmoid. [Online]. Available: https://www.sigmoid.com/blogs/how-to-detect-and-overcome-model-drift-in-mlops/