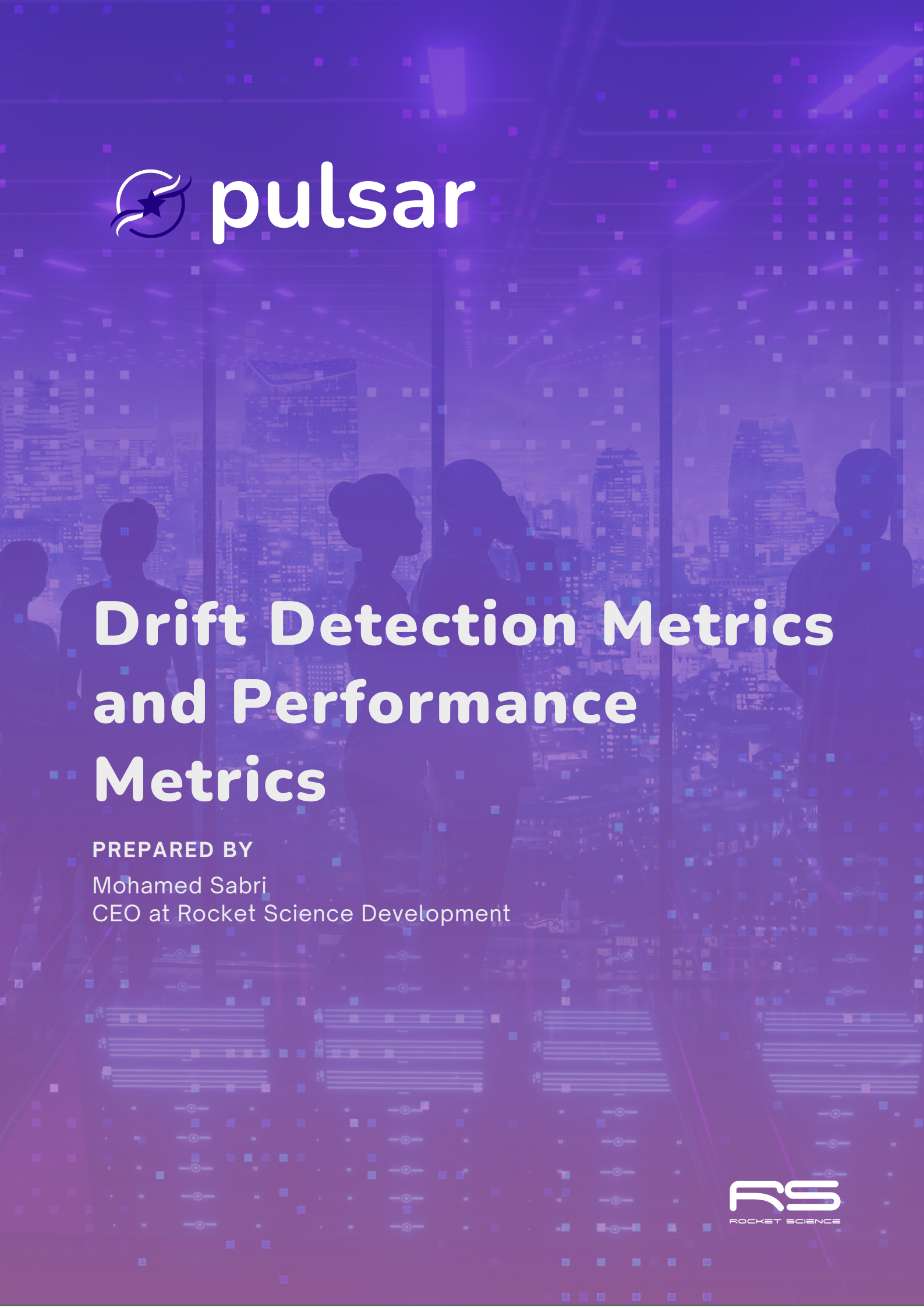


pulsar

Drift Detection Metrics and Performance Metrics

PREPARED BY

Mohamed Sabri
CEO at Rocket Science Development

ROCKET SCIENCE

pulsar

Drift and Performance in production, what is it about?

Drift detection metrics and performance metrics are two essential concepts in the field of machine learning, particularly in production environments where models are continuously updated and deployed. Drift detection metrics refer to the process of monitoring the performance of a deployed model over time and detecting any changes in its behavior. Drift detection can help identify situations where the data distribution has changed, or the model's accuracy has deteriorated, leading to a potential need for retraining the model. Some commonly used drift detection metrics include accuracy, precision, recall, and F1 score.

Performance metrics, on the other hand, are used to measure the effectiveness of a machine learning model. Performance metrics are typically used during the development and testing phase to assess how well the model performs on a given dataset. Some commonly used performance metrics include accuracy, precision, recall, and F1 score, as well as more specialized metrics such as AUC-ROC, mean absolute error, and mean squared error.

Pulsar.ml is a Python library that provides a range of metrics for measuring the performance of machine learning models in production environments. The library offers a variety of metrics for drift detection, including precision and recall metrics for classification problems, as well as metrics such as MAE (Mean Absolute Error) and MSE (Mean Squared Error) for regression problems. In addition to drift detection metrics, Pulsar.ml also provides a range of performance metrics for classification and regression problems, such as accuracy, F1 score, AUC-ROC, and mean squared error. These metrics can be used to evaluate the performance of a model during development and testing and monitor its performance in production environments. Here is the link to the Github: https://github.com/Rocket-Science-Development/pulsar_metrics and website https://pulsar.ml/

pulsar

Drift detection in production, what type of drift?

Drift detection in production is an essential aspect of maintaining the performance of machine learning models over time. There are several metrics that can be used to detect drift in production systems, and some of the popular ones are:

- **Concept Drift:** This occurs when the underlying concept or relationship between the input features and output labels changes over time. For example, a model trained to predict stock prices may experience concept drift if market conditions change significantly.
- **Data Drift**: This occurs when the statistical properties of the input data change over time. For example, a model trained on data from one geographic region may experience data drift if it is applied to data from a different region.
- **Covariate Shift**: This occurs when the distribution of the input data changes over time, but the relationship between the input features and output labels remains the same. For example, a model trained on data from a certain time period may experience a covariate shift if it is applied to data from a different time period.

pulsar

What are the tests used for drift detection?

To detect these types of drift, there are several metrics that can be used:

- **Kolmogorov-Smirnov (KS) Test**: This test compares the distribution of input data at different points in time to detect if there are any significant differences. If the p-value of the KS test is below a certain threshold, this may be an indication of drift.

- **Drift Detection Metrics (DDM)**: This is a set of metrics specifically designed for drift detection. DDM uses statistical tests to compare the performance of a model on new data with the performance on previously seen data. If the performance drops below a certain threshold, this may be an indication of drift.

- **Page-Hinkley Test**: This test compares the cumulative sum of errors generated by a model over time. If the sum of errors exceeds a certain threshold, this may be an indication of drift.

- **Concept Drift Detection (CDD) Metrics**: These metrics are designed to specifically detect concept drift. CDD uses statistical tests to compare the relationship between input features and output labels on new data with the relationship on previously seen data. If the relationship changes significantly, this may be an indication of concept drift.

Kolmogorov-Smirnov (KS) test

One of the most popular tests is still the Kolmogorov-Smirnov (KS) test is a statistical test that can be used to detect changes in the distribution of input data over time. It's a non-parametric test, which means that it makes no assumptions about the underlying distribution of the data.

The KS test works by comparing the empirical cumulative distribution function (ECDF) of the input data at two different points in time. The ECDF is a step function that represents the proportion of observations in the data that are less than or equal to a given value. By comparing the ECDFs at two different points in time, we can test whether there is a statistically significant difference in the distribution of the data.

Here's how the KS test works:

1. We start with two sets of input data: one set from the current time period (t) and another set from a previous time period (t-1).
2. We compute the ECDFs of both sets of data.
3. We calculate the maximum vertical distance between the two ECDFs. This distance is called the KS statistic (D).
4. We compare the KS statistic to a critical value from a pre-defined significance level and sample size. If the KS statistic is greater than the critical value, we reject the null hypothesis that the two sets of data come from the same distribution. This suggests that there has been a significant change in the distribution of the input data, which could be an indication of drift.

It's important to note that the KS test assumes that the two sets of data are independent and identically distributed (iid). This means that the data should be sampled from the same population and the samples should not be dependent on each other. If these assumptions are not met, the KS test may not be an appropriate method for detecting drift.

The performance in production and its metrics

Measuring the performance of ML models in production is crucial to ensure that they are functioning properly and delivering the desired outcomes.

pulsar

Here are some best practices for measuring the performance of ML models in production:

- **Define appropriate performance metrics**: The performance metrics used in training the model may not be appropriate for measuring its performance in production. Therefore, it's important to define appropriate metrics that reflect the business objectives and the impact of the model on the end users. Some common performance metrics for ML models in production include accuracy, precision, recall, F1 score, ROC curve, AUC, and others.
- **Monitor model performance in real-time**: Monitoring model performance in real-time can help identify issues as soon as they occur and facilitate prompt action to rectify the situation. This can be achieved by setting up performance dashboards that track key metrics and provide alerts when performance falls outside acceptable ranges.
- **Use data drift detection methods**: As discussed earlier, data drift can have a significant impact on model performance in production. Therefore, it's important to use data drift detection methods to monitor changes in the input data distribution and detect any drift. This can help trigger model retraining or adjustments to ensure that the model continues to perform optimally.
- **Conduct regular model re-evaluations**: Models can become less effective over time as data distributions and user behavior change. Therefore, it's important to conduct regular model re-evaluations to ensure that the model is still providing value to end-users. This can involve testing the model on new data and validating its performance against the defined metrics.
- **Evaluate model explainability**: Explainability is becoming increasingly important as ML models are used in more critical applications. It's important to evaluate the model's explainability to ensure that it can be understood and trusted by end-users. This can involve using explainability techniques such as feature importance analysis, counterfactual analysis, and others.

pulsar

Pulsar.ml: drift detection in real time

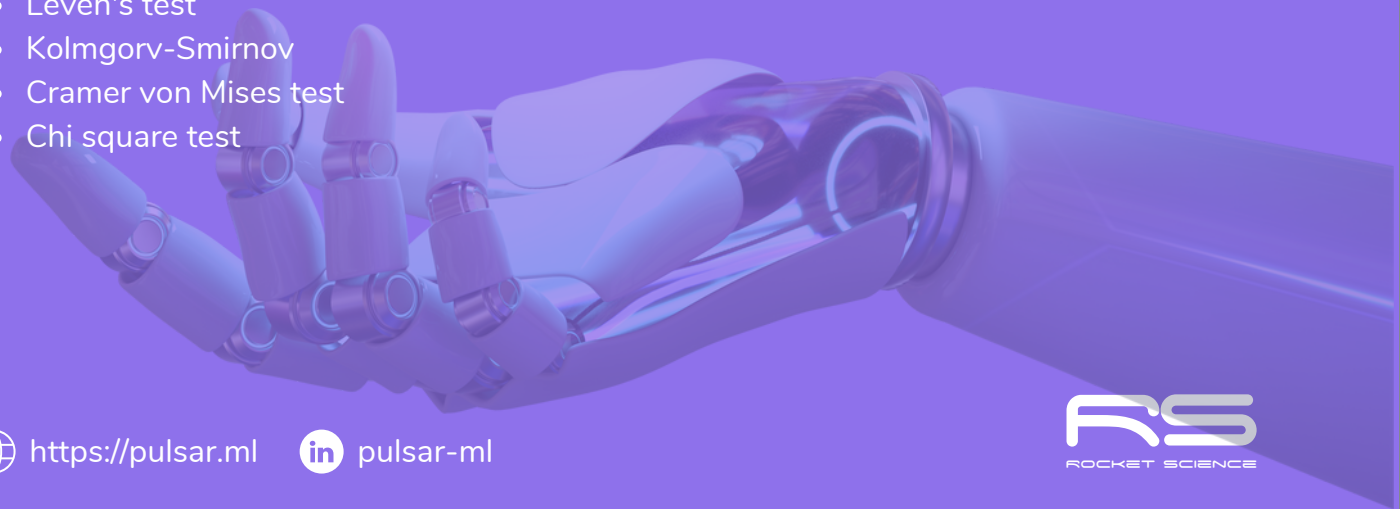
Pulsar.ml is a Python library for monitoring machine learning models in production. It provides a set of metrics for evaluating the performance of models and detecting data drift. Pulsar.ml can be used with any machine learning framework, including TensorFlow, PyTorch, and Scikit-learn.

Pulsar.ml includes the following features:

- **Model performance metrics**: Pulsar.ml provides a range of metrics for evaluating the performance of machine learning models. These metrics include accuracy, precision, recall, F1 score, ROC curve, AUC, confusion matrix, and others.
- **Data drift detection**: Pulsar.ml includes methods for detecting data drift in real-time. It compares the current data distribution to a reference distribution and generates alerts when significant changes are detected.
- **Support for multiple models**: Pulsar.ml can be used to monitor multiple machine learning models simultaneously. It provides a unified dashboard for visualizing the performance of all models in real-time.
- **Customizable alerts**: Pulsar.ml allows users to customize alerts based on their specific needs. Users can define thresholds for each metric and receive alerts when performance falls outside these thresholds.
- **Integration with other tools**: Pulsar.ml can be integrated with other monitoring and logging tools, such as Prometheus, Grafana, and Elasticsearch.

Some of the metrics supported by Pulsar.ml include:

- Metric
- Kullback-Leibler divergence
- Wasserstein distance
- T-test
- Mann Whitney U test
- Leven's test
- Kolmgorv-Smirnov
- Cramer von Mises test
- Chi square test



https://pulsar.ml pulsar-ml

ROCKET SCIENCE