Journal Homepage: -www.journalijar.com

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH (IJAR)

Article DOI:10.21474/IJAR01/18684
DOI URL: http://dx.doi.org/10.21474/IJAR01/18684

RESEARCH ARTICLE

MODEL MONITORING WITH GRAFANA AND DYNATRACE: A COMPREHENSIVE FRAMEWORK FOR ENSURING ML MODEL PERFORMANCE

Vinod Menon[1], Joemon Jesudas[2] and Gopika S.B[3]

1. Senior Data Engineer, Insurance Analytics & Researcher.
2. Data Engineer, Insurance Analytics & Researcher.
3. Data Scientist, Insurance Analytics & Researcher.

Manuscript Info

Manuscript History
Received: 05 March 2024
Final Accepted: 09 April 2024
Published: May 2024

Key words:-
Model monitoring, Machine Learning, Drift Analysis, Monitoring Tools, Grafana, Dynatrace, Model Performance Evaluation

Abstract

In the rapidly evolving landscape of Machine Learning (ML), ensuring the continued efficacy and reliability of deployed models is paramount. This paper introduces a robust framework for model monitoring, highlighting the capabilities of Grafana and Dynatrace. Grafana provides powerful visualization and alerting capabilities, while Dynatrace offers deep insights into system and application performance.By integrating these tools, organization can establish a holistic monitoring solution that tracks key performance indicators (KPIs), detects anomalies, and triggers alerts in real-time. This framework enables proactive management of model drift, data quality issues and resource utilization, thereby bolstering the trustworthiness of ML applications in production.Case studies and practical implementation guidelines illustrate the effectiveness of this approach across various industry domains. The proposed methodology represents a pivotal advancement in the field of ML model operations, facilitating enhanced decision-making, reduced downtime and heightened user satisfaction.

Introduction:-

"Finally, it's live!!" This is a huge milestone that an ML development team looks forward to where they have managed to deploy the ML model in production, however the story doesn't end here as a model deployed to production is not the final step of the machine learning workflow. As ML models operate in dynamic environments, they are susceptible to deviation from their excepted behaviour due to changing data distribution or data quality degradation which leads to inaccurate predictions. There can also arise resource bottlenecks that hinder optimal model operation. Addressing these issues in real-time is critical to maintaining user satisfaction, reducing downtimeand achieving regulatory compliance. The key goals achieved through effective model monitoring are:

1. Determining the accuracy of model predictions and determining whether the performance has decayed and if retraining is required or not.
2. Evaluation and testing pipelines represented by log metrics, charts, predictions distribution and confusion matrix.
3. Model training and retraining monitoring for achieving continuous high performance.

Corresponding Author:- Joemon Jesudas
Address:- Data Engineer, Insurance Analytics & Researcher.

Traditional monitoring solutions often fall short in providing comprehensive insights and proactive capabilities needed to manage ML models effectively. They lack the ability to offer end-to-end visibility into an ML model's complete journey, from data ingestion and training to inference. Additionally, these tools often lack real-time anomaly detection and alerting mechanisms, leaving organizations reactive rather than proactive in addressing issues.To meet these challenges head-on, organizations require a robust framework for model monitoring that bridge the gap between model development and production deployment. The proposed framework leverages the capabilities of industry-leading tools such as Grafana and Dynatrace, to establish a holistic monitoring solution and understand the extent of usability of various tools depending on the use-case.This paper introduces such a use case, showcasing how the integration of Grafana and Dynatrace provides a comprehensive solution for tracking key performance indicator(KPIs), detecting anomalies, analyzing model drift and triggering real-time alerts in ML model deployments. By doing so, organization can proactively manage model drift, data quality, and resource utilization, ultimately heightening the reliability of ML applications in production environments.

In the following sections, we delve into the components, benefits and practical implementation of this integrated monitoring framework, providing insights and guidance that showcase its effectiveness across various industry domains.

Literature Review:-

In recent years, the proliferation of artificial intelligence (AI) and machine learning (ML) technologies has ushered in a new era of data-driven decision-making for organizations across various industries. The ability to leverage data to gain insights, make predictions, and automate processes has become a competitive advantage in today's fast-paced business landscape. However, the widespread adoption of AI and ML has brought to the forefront a set of unique challenges related to model performance, data quality, and real-time operational management.

1. **Importance of model monitoring**

The ML models are dynamic and their performance degrade over time once deployed in production. Monitoring machine learning(ML) models is of paramount importance in the life cycle of AI/ML and data-driven applications. ML Model monitoring provide the capability to detect anomalies in real-time. In machine learning (ML) operations, feature monitoring plays a crucial role in ensuring the accuracy, reliability, and effectiveness of the models. When teams continuously monitor the features used in their predictive models, they can make informed decisions, detect anomalies, and maintain high-quality predictions. Monitoring your ML system is a necessity once it is deployed.

Effective feature monitoring hence offers several benefits, such as improving your model's performance, early alert systems for proactive team response, enhanced data quality, continuous model improvement, and efficient resource allocation. These naturally lead to better decision-making and customer outcomes.

Here are some several key reasons why model monitoring is crucial:

a) Early Detection of Anomalies

Anomalies may indicate issues such as data quality problems. Early detection of anomalies ensures the continued reliability and accuracy of ML models in production environment. Anomalies can be analysed by residual analysis, concept drift detection, Feature importance changes and etc…

b) Performance Assurance

Monitoring allows data scientist or machine learning engineer to ensure that ML models perform as expected in production environment. Detecting and addressing performance issues, ensures that the model's prediction remains accurate over time. Thus, optimizing a model helps achieve better performance on the target task, such as higher accuracy, lower error rate, or better prediction quality, resulting in more effective and reliable outcomes.

c) Data Quality Assurance

Model monitoring can identify issues with data quality such as outliers, incorrect data etc. Ensuring data quality is paramount because models are highly dependent on the quality of the data they are trained on and make predictions with. A well-optimized model can better generalize to unseen data, reducing the risk of overfitting (when a model performs well on the training data but poorly on new data) and ensuring higher out of sample accuracy. Thus model monitoring drives to well-optimized models by identifying and improving data quality.

d) Enhanced User Experience

Monitoring helps maintain the quality of customer experiences by ensuring that recommendations, personalization, and predictions remain accurate and relevant. Any fluctuations notified can be addressed at the earliest , thereby reducing the downtime and provide higher customer satisfaction.

e) Data-Driven Decision-Making

As ML model monitoring can help by:

1. Providing insights into model behaviour and model performance in production.
2. Alerting when issues arise e.g. concept drift, data drift, or data quality issues.
3. Providing actionable information to investigate and remediate these issues.
4. Providing insights into why your model is making certain predictions and how to improve predictions.

These insights allow data scientists and ML teams to identify the root cause of problems, and decide precisely on how to evolve and update models to improve accuracy in production. Monitoring insights are therefore crucial in deciding the model retraining and refinements.

f) Challenges in ML Model Monitoring

While AI and ML offer immense potential, they are not without their challenges. One of the most critical challenges is the continuous monitoring and management of ML models in real-world, dynamic environments. These challenges include:

a) **Concept Drift** : This is one of the most significant challenges faced in model monitoring. Over time the incoming data distribution may change, rendering the model less accurate. So Detecting and adapting to concept drift is essential to maintain model performance.When a model initially performs well in production but then degrades in performance over time, this indicates drift. ML monitoring tools or observability for machine learning systems can help you detect this drift, and take actions on tracking performance metrics, identify how it affects the model, and make recommendations for improving it. Traditional solution may struggle to detect concept drift (change in data drift) promptly, which is essential for maintaining model accuracy in dynamic environment.Here are three ways to prevent concept drift:
Model monitoring – reveals degradation in model performance that could indicateconcept drift, thus prompting ML developers to update the model.
1)Time-based approach – Determining a degrade timeframe for retraining. For example, if model performance becomes unacceptable every four months, retrain every three months.
2)Online learning –Performance monitoring helps us detect that a machine learning model in production is underperforming and understand it's cause. This often includes monitoring model activity, metric change, model staleness (or freshness), and performance degradation. The insights gained through model performance monitoring will advise changes for improving performance, such as hyperparameter tuning, transfer learning, model retraining, developing a new model, and more.

b) **Data Quality:** Poor data quality leads to issues in model monitoring. Inaccurate or incomplete data can result in false alarms or missed issues, affecting the model reliability. In practice, data scientists understand that the fact that the model performance will slightly differ on real-world data as compared to the test data used during development. In addition to that, real-world data is very likely to change over time. For these reasons, we can expect and tolerate some level of performance decay once the model is deployed. To this end, we set an upper and lower bound for the expected performance of the model. The data science team should carefully choose the parameters that define expected performance in collaboration with subject matter experts.

c) **Model Decay:** Even well-trained models can degrade in performance over time if not monitored and retrained. Keeping track of model accuracy and knowing when and how to retrain the model is essential. This can help you identify a need for better model environments or more computational resources. Model drift refers to the change in the statistical properties of the target function that a machine learning model is trying to approximate. As the distribution of the data changes over time, a mismatch between the training data and the test data can occur. Even successful models can fail to meet end-user expectations if they are too slow to respond. Thus ML Monitoring tools can help you identify if a prediction service experiences high latency, and why different models have different latency.

d) **Historical Data:** Monitoring requires access to historical data to establish baselines and detect anomalies. Ensuring that historical data is available and maintained can be a challenges. High volumes of historical data and its scarcity for particular use case can be a challenge in model monitoring. In scenarios with high data volumes, traditional monitoring system may struggle to process and analyse the data efficiently leading to delay and increased computational requirements.

e) **Scalability and Real-Time Monitoring:** Some machine learning models require real-time monitoring, which adds complexity. Ensuring that monitoring systems can handle real-time data and scale as needed is challenge. Traditional model monitoring often operates on batch processes, meaning that models are evaluated

periodically rather than in real-time. This delay can result in issues going undetected until the next monitoring cycle.

f) **Limited to model Metrices:**Traditional model monitoring solutions typically focus on monitoring model-specific metrices(e.g. accuracy, precision,recall) but may not provide insight into broader context of model such as data quality issues or feature importance.

g) **Lack of Anomaly Detection:**Many traditional monitoring tools focus on evaluating models against predefined thresholds and may not have the capabilities to detect unexpected model behaviour.

h) **Complex configuration**:Setting up and configuring traditional monitoring solutions can be complex and time-consuming, requiring a deep understanding of both the model and monitoring tool.

i) **Difficulty in explaining model behaviours:**Traditional solutions may not provide adequate information or capabilities for explaining why a model make specific prediction, limiting their usefulness for debugging and root-cause analysis.

Methodology:-

In the pursuit of robust and effective model monitoring solutions, the proposed framework aims to undertake a comprehensive evaluation of two prominent platforms, Grafana and Dynatrace. This evaluation seeks to determine their suitability and performance in the context of model monitoring, a critical aspect of modern data-driven application and machine learning systems.

Overview:

a. **Grafana**

Grafana is an open-source observability and data visualization platform designed to help organizations monitor, analyse, and understand their metrics, logs, and other data sources in real-time. It is also known for its flexibility and extensibility making it promising candidate for model monitoring. Grafana supports a wide variety of data source such as Prometheus, OpenTSDB and InfluxDB as well as SQL databases like MySQL and Postgres. Regardless of the data's storage location, it enables you to perform data queries through the query editor, create visual representations using dashboards, and set up alerts using the alerting functionality.

b.**Dynatrace**

Dynatrace is a leading observability and application performance management (APM) platform designed to provide comprehensive insights into the performance, availability, and behaviour of applications and infrastructure. It is widely used by organizations to monitor and optimize their digital ecosystems, ensuring that applications run smoothly, perform efficiently, and deliver an exceptional user experience.

While Grafana visualize time-series data from multiple sources and create AI/ML model monitoring systems and application performance, Dynatrace encompasses Activegate custom extension which allows us to run queries against different databases and send results back to Dynatrace as metrics. This extension helps to bring in business data, job processing metrics, database and table statistics, anything that is stored in your Databases.

Implementation and Experimental Results:-

Architecture:-

The below architecture is an overview of the machine learning (ML) lifecycle stages, including development, deployment, validation and testing. Initially, the model gets trained on historical data and tested before being implemented as a prediction service in a production environment. The feature inputs from Client to the prediction service generates the corresponding predictions from the model's end which are transferred back to the client. The execution metadata from the prediction service is directed to the PostgreSQL through kafka which accelerates the data retrieval process. The reference data from the development environment are also directed to the model monitoring system . The output from model monitoring system redirects the way in which the model is to be retrained, considering the model drifts and anomalies thereby improving the overall performance and accuracy of the prediction system.

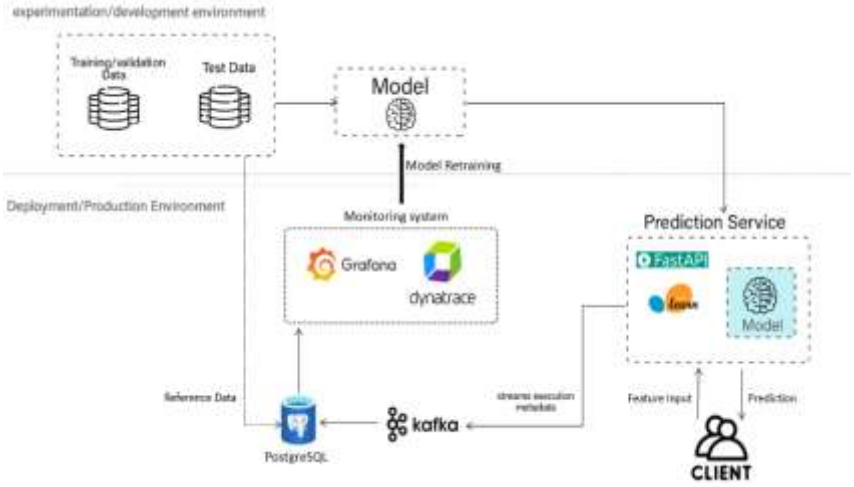

Fig 1:- Model monitoring architecture.

Implementation Using Grafana

Once the data has been streamed from the prediction service to the Postgres DB, a direction connection can be established between Grafana and Postgres DB, since Grafana supports a wide verity of data sources include Postgres.

To configure the data source, the below steps needs to be followed,

a. Click connection in the left-side menu
b. User connection, click Data sources and select PostgreSQL from the list.
c. And set the basic data source configuration like:

1. Name of the Connection.
2. Host address
3. Database
4. Username
5. Password
6. Port number

Once the above configuration is made, the connection with the database will be established and with the help of SQL queries we can retrieve the data stored in the table to the Grafana dashboard.

Results:-

The model monitoring dashboards are created for a sample insurance domain specific prediction model. The KPI's will include terms such as claimed amount, Total Savings, Market value and model prediction.

1**) Model Monitoring Dashboard Using Grafana**

Fig 2:- Model Monitoring showing Claimed amount and prediction drift.

The Fig 2 depicts the visualization of prediction drift over time. The purple straight line shows the trained dataset average and blue base line shows the prediction average respectively. The yellow trend shows the drift in model prediction while the green line depicts the feature drift.

Fig 3:- Model Monitoring showing data drift.

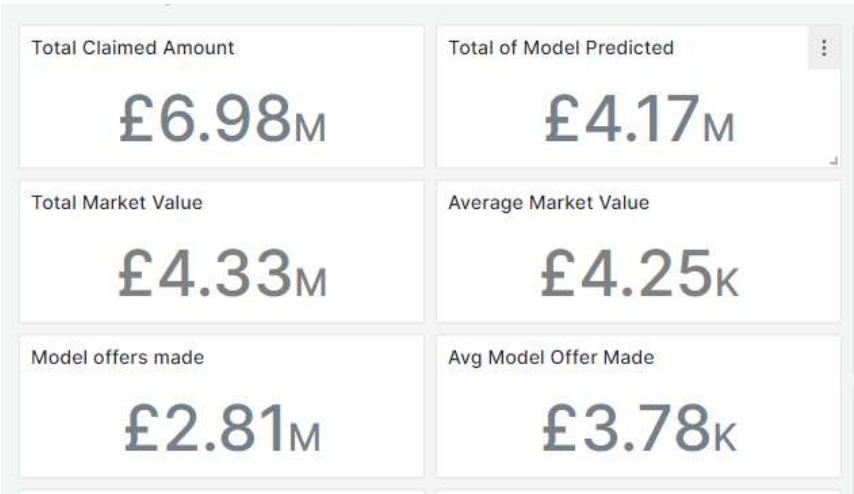

Fig 4:- Model monitoring showing KPI's.

The above Fig 4 corresponds to the KPI's that are relevant in monitoring the model. These KPIs refer to a set of quantifiable measurements used to gauge a model's overall long-term performance.

Fig 5:- Model monitoring dashboard showing model performance.

The above Fig 5 depicts the prediction performance with respect to claimed amount in different claimed amount buckets. This shows an overall performance of the model in each bucket.

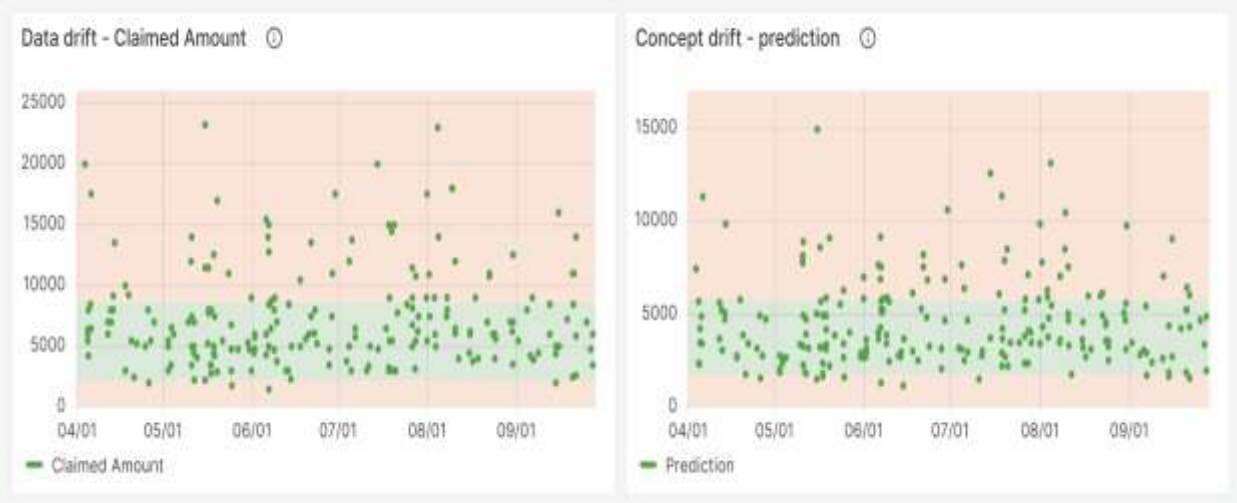

Fig 6:- Model monitoring dashboard showing anomalies detection.

The Fig 6 visualize the anomalies detection against the actual data and trained data. The green bandwidth in data drift represents the threshold range for trained dataset whereas the right hand side represents the threshold range for prediction on trained data. As upper and lower bound of the threshold range indicates the sum and difference of the mean and standard deviation, this shows the range of distribution of the data over time.

2)Implementation Using Dynatrace

Dynatrace supports PostgreSQL database connectivity directly for application monitoring and services but doesn't have the capability for monitoring machine learning models. However, Dynatrace supports activegate custom extension that allow you to run queries against Oracle, MSSQL**,** MySQL, DB2, PostgreSQL, Informix or SAP HANA databases and send the results back to Dynatrace as metrics. Using this extension we can bring in business data, job processing metrics, database and table statistics, anything that is stored in your Databases.

This extension utilizes JDBC to connect to the different databases, for example for Postgres, supporting JDBC driver should be downloaded and deployed manually.

The custom Database queries extension can be downloaded from Dynatrace hub and once downloaded it should be uploaded manually. For that, steps include: Navigate settings -> Monitored Technologies -> custom extension -> Upload extension.

Once the extension is successfully uploaded, the endpoint can be seen in settings -> Monitored technologies > Custom extensions > Generic DB Query Plugin

Upto 10 queries can be run at a time, and to run more than 10 queries, create more endpoints for the same database. The Parameters for this endpoint include:

1. Dynatrace API Token: The API Token, it needs Api V2 - Metric Ingest permissions
2. Database Type : Oracle, MSSQL, MySQL, DB2, PostgreSQL and Informix are supported
3. Hostname
4. Port
5. Username
6. Password
7. Query name(1-10)
8. Query String(1-10)
9. (Optional) Query (1-10) – Schedule - The schedule, it uses cron format to schedule the query. If empty, the query will run every minute.

Once the parameters are provided and configured, the data will be send using Metric Ingest API. With the help of Data Explorer tab metrices can be integrated to the dashboard. The metric name is always custom.db.query, the different queries are accessed via the Query Name dimension.

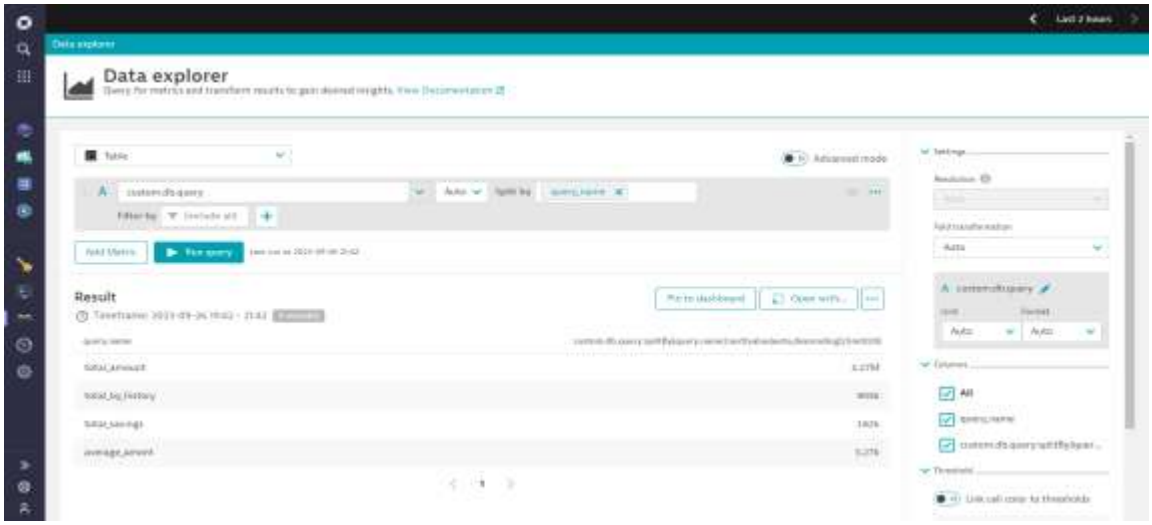

Fig 7:- Data explorer tab – The metrics returning tab.

Results:-

Model Monitoring Dashboard Using Dynatrace

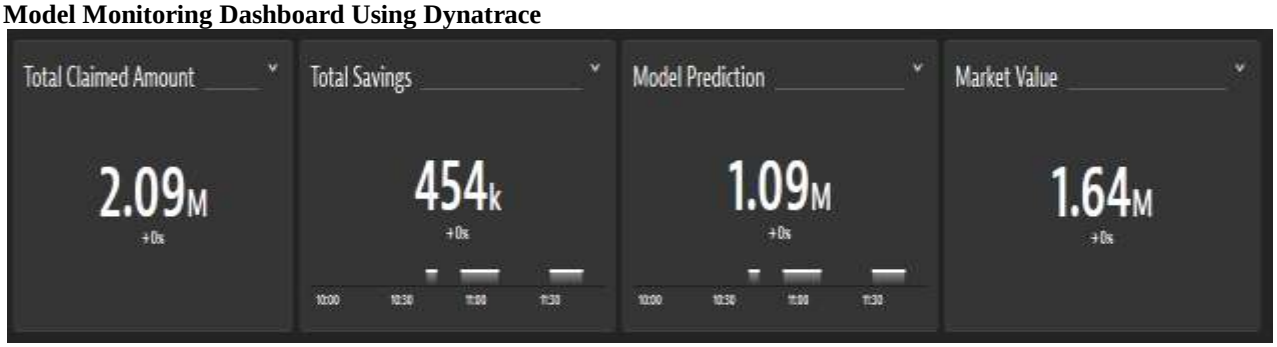

Fig 8:- Model monitoring Dashboard showing Different KPI's.

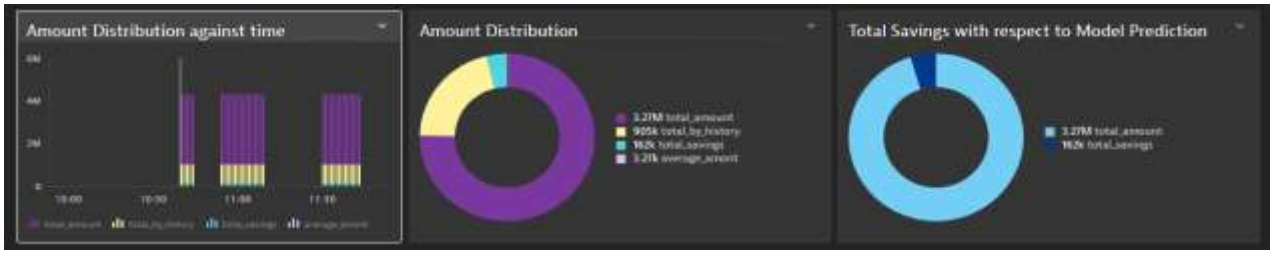

Fig 9:- Model Monitoring Dashboard showing Data distribution stats.

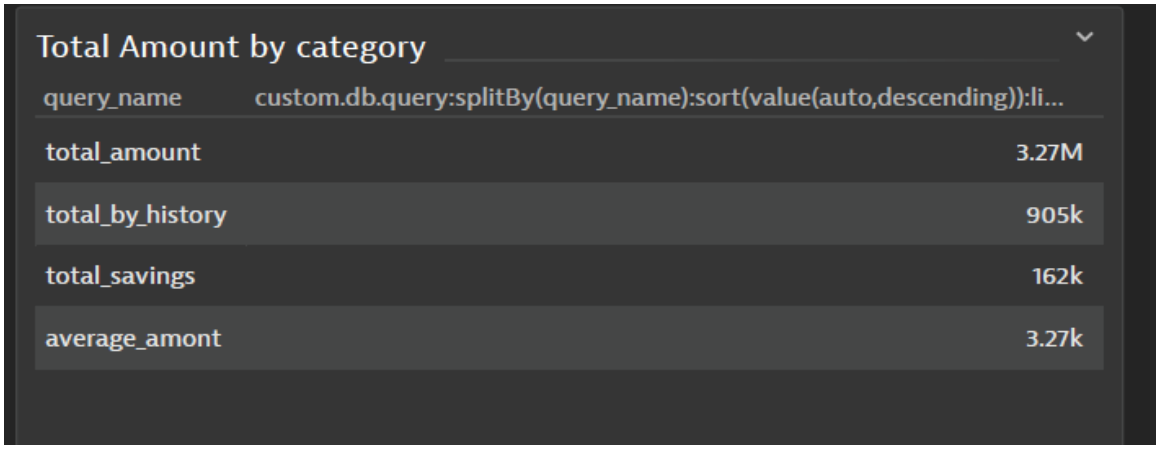

Fig 10:- Model Monitoring Dashboard showing Reference Data stats.

Conclusion and Future Work:-

In the realm of model monitoring, the choice between Dynatrace and Grafana depends on specific use cases, requirements and priorities. Both tools offers valuable capabilities, but they come with their own strengths and limitations. Dynatrace excels in its comprehensive approach to application monitoring, which includes model monitoring as part of its broader observability suites. Its integration with data source is limited and have limited functionalities. The result of every query is returned as a metrics and only single number will be returned. This particularly makes it well suited for monitoring overall behaviour and application performance. Additionally, Dynatrace's alerting and customizable dashboards help in proactive issue identification and resolution.

However, Dynatrace may come with a higher cost and complexity compared to dedicated model monitoring tools. Its Primary focus is on application performance monitoring , while it can monitor ML models, this might not offer the same level of model-specific features and visualizations as provided by dedicated tools like Grafana.

On the other side, Grafana is a versatile open-source tool often used for creating custom dashboards and visualizations. Its strength lies in its flexibility, extensibility and support for various data sources, including model monitoring. Grafana allows you to build tailored dashboards that specifically focus on model's performance metrics, making it a great choice if the primary concern is model monitoring.

However, Grafana's capabilities for AI Driven anomaly detection and automated alerting may be limited compared to Dynatrace. You need to rely on external tools or plugins to implement these features effectively.

Future scope:Consider using both Dynatrace and Grafana in a hybrid approach. Dynatrace can provide overall system monitoring and alerting, while Grafana can be used to create customized dashboards specifically tailored to model performance. Integrating this hybrid approach can provide a comprehensive solution for models in dynamic environment.

References:-

1. Grafana Labs (2023) The analytics platform for all your metrics. https://grafana.com/
2. S. Shankar, A. Parameswara, Towards Observability for Machine Learning Pipelines, 2021, https://arxiv.org/abs/2108.13557.
3. P. Baier, S. Dragiev, Challenges in Live Monitoring of Machine Learning Systems, The Upper-Rhine Artificial Intelligence Symposium UR-AI 2021, ARTIFICIAL INTELLIGENCE - APPLICATION IN LIFE SCIENCES AND BEYOND, 2021.
4. Y. Wu, E.D. Yinjun, S.B. Davidson. DeltaGrad: Rapid retraining of machine learning models, International Conference on Machine Learning, 2020.
5. Dynatrace The analytics and automation platform powered by causal AI. https://dynatrace.com/