Monitoring ML models on-premise requires tools that can provide insights into model performance, data quality, and system health while respecting the infrastructure constraints and data privacy requirements of on-premise deployments. Here are some options for effective ML model monitoring on-premise:

**1. Prometheus and Grafana**

* **Prometheus**: A powerful open-source monitoring and alerting toolkit. Prometheus can scrape metrics from various sources and store them for querying. It’s widely used for infrastructure monitoring and can track model performance metrics when paired with model-serving software that exposes such metrics.
* **Grafana**: Works well with Prometheus to visualize the data, allowing you to build custom dashboards for metrics like latency, response rates, and prediction distributions.
* **Best For**: Real-time monitoring, customizable dashboards, and alerting on system metrics and model latency.

**2. ELK Stack (Elasticsearch, Logstash, Kibana)**

* **Elasticsearch**: Used for storing and searching log and metric data.
* **Logstash**: Collects and transforms data from various sources for Elasticsearch.
* **Kibana**: Provides data visualization for Elasticsearch data, offering dashboards to analyze logs, model prediction errors, or other performance metrics.
* **Best For**: Logging-based monitoring and tracking for both model performance and infrastructure, especially when text and event data are involved.

**3. Apache Kafka and Apache Spark**

* **Apache Kafka**: Handles data streaming and can collect logs, metrics, and events from model inferences.
* **Apache Spark**: Performs analytics on streaming data from Kafka, which can then be used to identify performance issues or shifts in model behavior in near real-time.
* **Best For**: Scalable data pipelines for high-frequency model monitoring and detecting anomalies over large datasets.

**4. MLflow**

* MLflow can be configured on-premise and provides end-to-end tracking and versioning of ML models. While it’s primarily for experiment tracking, it also offers model monitoring capabilities and can log metrics such as accuracy, loss, or any other custom metrics.
* **Best For**: Comprehensive tracking of model metrics and monitoring, especially in environments that already use MLflow for experiment management.

**5. Seldon Core with Alibi Detect**

* **Seldon Core**: An open-source platform for deploying and managing machine learning models on Kubernetes.
* **Alibi Detect**: An open-source library that integrates with Seldon Core for detecting data drift, adversarial instances, and outliers.
* **Best For**: Kubernetes environments where models need to be monitored for both deployment and behavior, especially for detecting data quality and drift.

**6. DataRobot MLOps**

* DataRobot offers an on-premise solution focused on model monitoring and performance tracking. It includes built-in tools for detecting data drift, outliers, and accuracy degradation.
* **Best For**: Organizations needing enterprise-level support and are willing to invest in commercial solutions tailored for on-premise deployments.

**7. ZenML**

* ZenML is an open-source MLOps framework that enables end-to-end workflows and can be set up on-premise with various integrations. It’s adaptable to multiple deployment and monitoring solutions like Grafana, Evidently AI, and Prometheus.
* **Best For**: Flexible pipeline and workflow monitoring when used with additional monitoring solutions, and especially when integrating multiple tools.

**8. Evidently AI**

* Evidently AI is an open-source tool designed for model performance monitoring, focusing on detecting concept drift, data drift, and monitoring feature distributions. It can be used independently or in conjunction with tools like Grafana and Prometheus.
* **Best For**: Specialized ML monitoring, particularly for detecting model drift and data issues, with support for on-premise deployment.

**9. Kedro + Kedro-Viz**

* Kedro is an open-source framework for data pipeline development. It doesn’t directly monitor ML models but can be used to log metrics and track pipeline performance with custom visualization setups.
* **Best For**: Teams using pipeline-first approaches to monitoring data quality and performance for ML models in conjunction with model-serving frameworks.

**Considerations for Choosing an On-Premise ML Monitoring System**

When choosing an ML monitoring system for on-premise deployments, consider factors like:

* **Data Privacy and Compliance**: Ensure compliance with any data handling regulations.
* **Scalability**: If you expect model monitoring to scale, pick solutions like Kafka or Prometheus that handle high loads.
* **Integration Requirements**: Some tools integrate better with certain orchestration frameworks (e.g., Kubernetes).
* **Complexity vs. Ease of Use**: Tools like MLflow and Seldon Core offer a balance of usability with monitoring features, while others might need more custom setup.

Each of these tools has strengths suited to different environments and monitoring needs, so the best fit often depends on the infrastructure in place and the specific ML monitoring requirements.