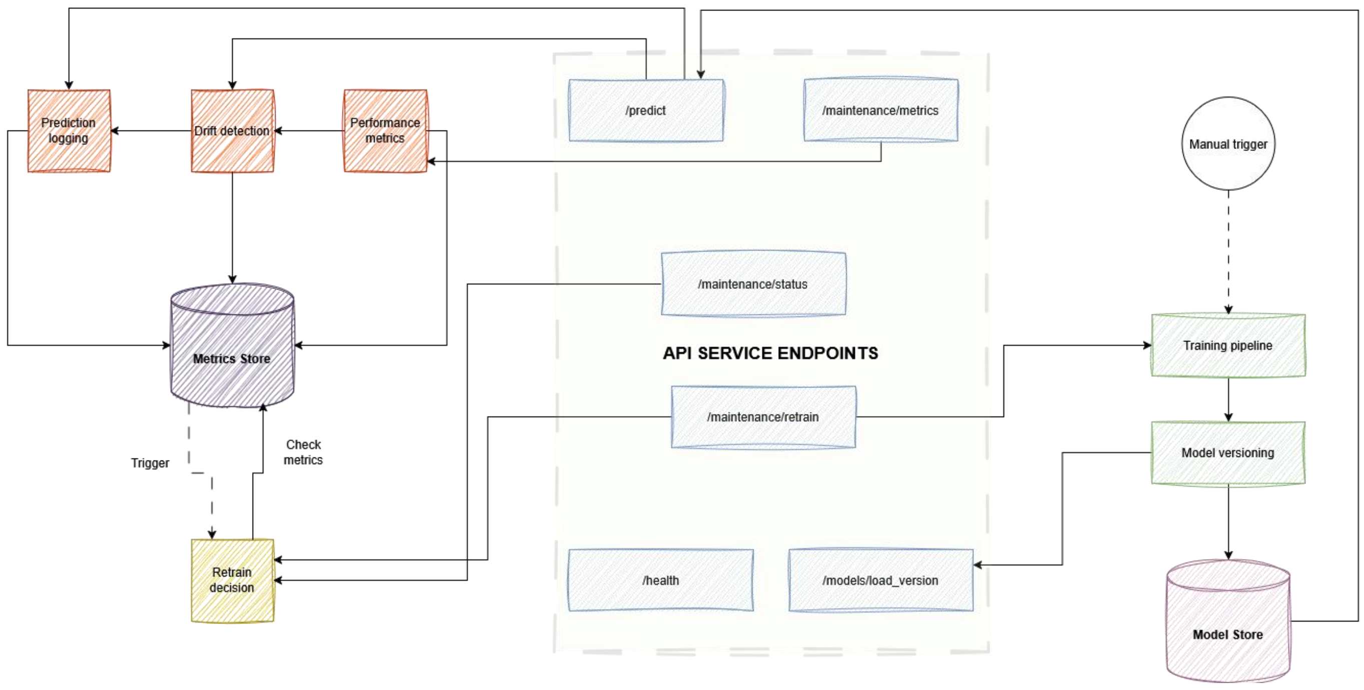
Monitoring & Maintenance

The production monitoring and maintenance parts implements an approach to model observability and automated maintenance. The architecture consists of three primary components: a model monitor, retrainer, and API service layer working together to ensure model reliability and performance in production.



Core components

The model monitor maintains a metrics store implemented as a JSON based system that tracks prediction history, performance metrics, and drift detection results. It retains a rolling window of the last 1000 predictions (could be decreased or increased), including response times, input features, and model versions. The monitor actively tracks average response times, 95th percentile latency, prediction rates, and feature distribution changes to detect potential model degradation.

The model retrainer implements a versioned approach to model updates. It maintains separate storage for training data and model artifacts, with each training run generating unique timestamps and metadata. Without any external libraries so the full observability and flexibility can be controlled on developer side. The retrainer only has one option as of now which is to retrain ensemble models instead of choosing between different approaches for simplicity. But it comes with a versioning system that allows for easy and live rollbacks.

The API endpoints exists for both inference and maintenance operations. The /predict endpoint integrates real-time monitoring, while maintenance endpoints expose system health metrics and control model versioning. The API implements automated retraining triggers based on configured thresholds for data drift and performance degradation.

Further, there is the option to do manual training by simply going to the code's root folder and running the following in the terminal ”`python main.py –version_id=[INSERT YOUR VERSION]`” .

Monitoring criteria

The system initiates retraining under the following conditions:

- Detection of data drift beyond configured statistical thresholds (mean, standard deviation)
- Average response time exceeding 1 second (on average)
- Significant degradation in model performance metrics
- Manual triggers through the maintenance API

Implementation details

The metrics store maintains structured data including:

- Prediction logs with timestamps and response times
- Model version metadata
- Performance metrics history
- Drift detection results

Model versioning implements:

- Timestamp based version control
- Training metadata preservation
- Ensemble model support
- Easy version switching through API endpoints