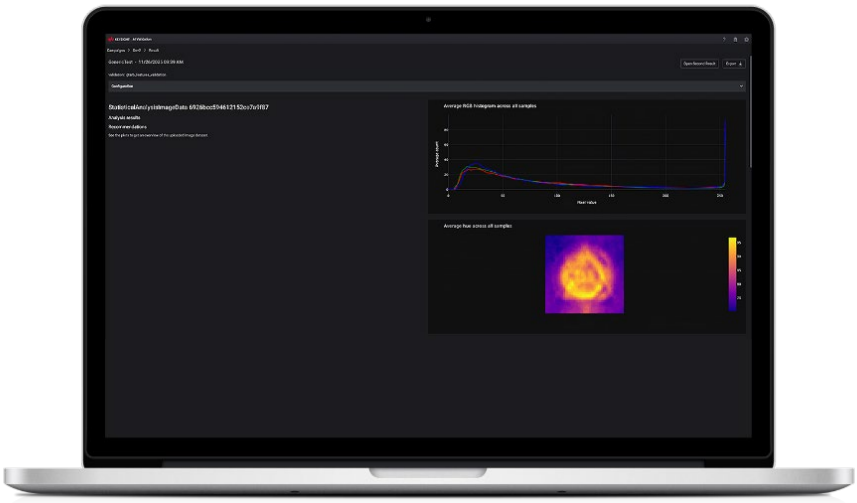
AX1000A
AI Software Integrity Builder

Designed for Compliance, Built for Scale – Holistic AI Assurance Across the Lifecycle

Software Solution for Holistic AI Safety and Conformance Evidence

The Keysight AI Software Integrity Builder introduces a lifecycle-based approach to AI Assurance, delivering an integrated framework for validating and maintaining AI-enabled systems in safety-critical domains such as automotive. By combining dataset analysis, model-based performance validation, and inference-based testing, it ensures transparent, reliable, and scalable AI validation to enable trustworthy AI deployment - aligned with emerging standards like ISO/PAS 8800 and regulatory requirements such as the EU AI Act.



KEYSIGHT

Table of Contents

AI Assurance Approached in a Lifecycle

The **AI Software Integrity Builder** introduces a novel, lifecycle-based approach to AI Assurance, designed to transform how AI-enabled systems are validated and maintained in safety-critical environments such as automotive. As AI development grows in complexity and regulatory pressure intensifies, this integrated solution delivers transparent, adaptable, and data-driven validation across the entire AI lifecycle.

AI systems operate as complex, dynamic entities, yet many of their decisions remain hidden from developers and users. This black-box nature and corresponding lack of transparency creates significant challenges for industries where safety and compliance are non-negotiable. Standards such as ISO/PAS 8800 and emerging regulations like the EU AI Act demand AI explainability and validation of safety in deployment but provide limited guidance on how to achieve this. Fragmented toolchains and siloed workflows further increase risk and effort, leaving gaps in regulatory conformance.

Keysight's AI Software Integrity Builder addresses these challenges by providing holistic AI Assurance capabilities, enabling teams to answer the critical question: **What happens inside my black box, and how do I ensure a trustworthy AI deployment?**

Integrated Software Framework Supporting AI Development & Maintenance

The solution helps to deliver the safety evidence required for regulatory conformance, empowering teams to validate, explain, adjust, and continuously improve AI systems. Unlike fragmented toolchains that address isolated aspects of AI testing, Keysight's integrated framework spans the entire lifecycle: from dataset analysis and model validation to inference-based testing in real-world environments.

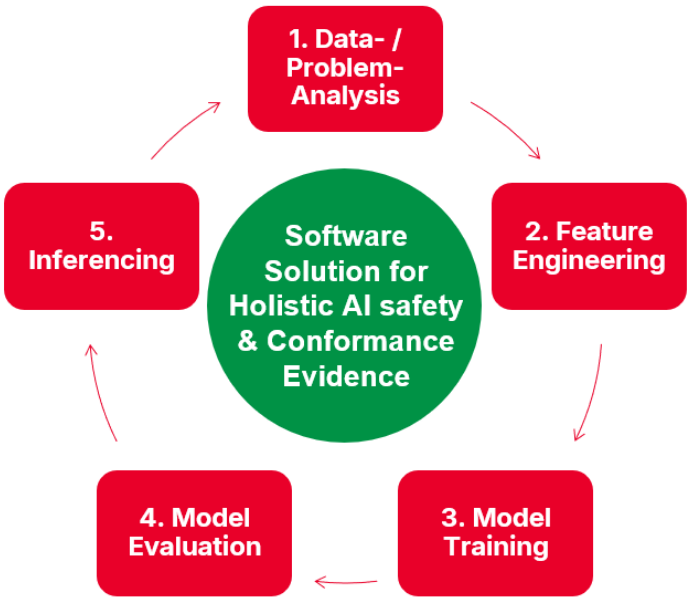


Figure 1. AI assurance approached in a lifecycle

Core capabilities include:

- **Dataset analysis:** Analyzes data quality using statistical methods to uncover biases, gaps, and inconsistencies that may affect model performance.
- **Model-based validation:** Explains model decisions and uncovers hidden correlations, enabling developers to understand the patterns and limitations of an AI System.
- **Inference-based testing:** Goes beyond static validation by analyzing how the model performs in real-world environments, detecting deviations from training behavior, and recommending improvements for the next iteration.

This unified approach ensures traceability, explainability, and compliance by design, empowering engineering teams to deploy AI systems that are not only performant but also auditable.

Stage 1: Data and Problem Analysis

The foundation of trustworthy AI begins with high-quality data and a clear problem definition. Keysight's AI Software Integrity Builder provides **statistical and semantic analysis tools** to identify biases, gaps, and inconsistencies in datasets. This ensures models are trained on representative and reliable data, reducing the risk of systemic errors later in the lifecycle.

Why it matters:

- For a trustworthy AI deployment, high-quality data is needed that is aligned with the Operational Design Domain (ODD) where the system will operate in.
- For this, it is important to
  - Define the problem.
  - Identify the data needed.
  - Analyze the data for bias.

Collecting the right data is not an easy task and requires both domain and AI expertise. We can help you identify gaps in your data set with the following main features:

- **Statistical analysis:** Detects anomalies, distribution shifts, and hidden biases. This includes evaluating dataset properties, such as missing values in tabular data and pixel-value distributions in image datasets.
- **Semantic checks:** Validates that the dataset aligns with the defined problem and operational context to ensure data relevance and data fit.
- **Gap identification:** Highlights missing or underrepresented datapoints

Figure 2 shows the **distribution of labels over a given dataset**. With this exemplary analysis, you can analyze the balance of your dataset. Analyzing label balance is critical for detecting class imbalances that can lead to biased models and poor generalization.

A **balanced distribution** ensures that all classes are adequately represented, reducing the risk of overfitting to dominant classes and improving fairness and robustness in predictions.

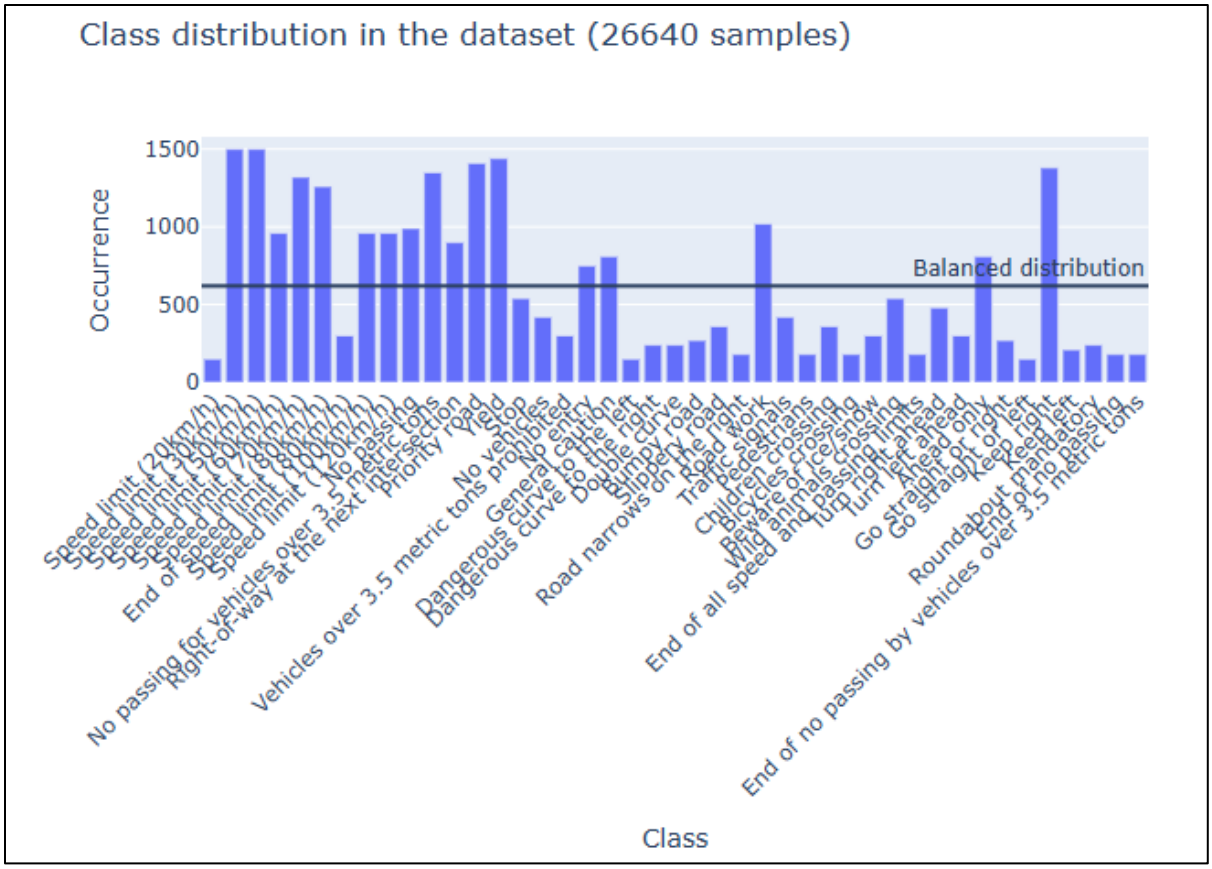


Figure 2. Example of a label distribution analysis

These methods from Stage 1 provide an essential foundation for trustworthy AI by highlighting gaps and **guiding data collection efforts**. With this, you are setting the stage for a high-quality dataset by identifying mislabeled data, imprecise or ambiguous labels, as well as underrepresented instances.

Stage 2: Feature Engineering

Feature selection and transformation are critical for model performance and interpretability. Keysight's AI Software Integrity Builder supports **feature analysis and correlation mapping**, helping developers understand which features drive predictions and where potential vulnerabilities may exist. This transparency is essential for compliance with explainability requirements.

Why it matters:

- Well-engineered features improve accuracy and reduce bias.
- Understanding feature influence supports AI explainability.
- Detecting spurious correlations prevents hidden dependencies and systemic risks.

To achieve these goals, our solution provides a set of key capabilities designed to ensure robust and interpretable feature engineering:

- **Feature scale evaluation:** Assesses normalization and scaling impacts on model behavior and ensures all features and target variables are correctly scaled.
- **Mutual information analysis:** Measures the influence of individual features on the target variable and captures nonlinear dependencies to reveal complex relationships. It also identifies features with low information content, ensuring data is relevant to the problem and reducing noise in the model.
- **Dataset distribution mapping:** Visualizes feature spread to uncover imbalances and overlapping distributions that may cause ambiguity.
- **Area of overlap detection:** Detects whether the feature spaces of different datasets are similar. A high area of overlap is desirable because it indicates that the validation dataset covers most regions of the training dataset, reducing blind spots and improving generalization.
- **Train/Test/Validation split analysis:** Verifies that subsets are well-sampled, similar but not identical, and free from contamination, which is critical for reliable testing and validation.
    - Contamination refers to data leakage between subsets, where information from the training set unintentionally appears in validation or test sets, leading to overly optimistic performance estimates.
    - In contrast to the above-mentioned analysis, *Area of overlap detection,* here too much overlap is undesirable, because it means the subsets are not sufficiently independent, which compromises the integrity of performance evaluation.
- **Impairment checks:** Evaluates the impact of transformations such as rotation, decals, and aging. While these techniques can boost robustness, they may introduce spurious correlations that need to be monitored.
- **Continuous quality assurance**: Mandatory after each transformation step to maintain data integrity and prevent unintended biases.

Figure 3 shows an **example analysis of the overlap between a training and a validation dataset**. In the plot, the red line represents the convex hull of the extracted features from the training dataset, while the green line represents the convex hull of the validation dataset.
The gap between the two hulls indicates regions where the machine learning (ML) model would not be validated if the validation dataset were used in this form, potentially creating blind spots. A high area of overlap (here indicated as 0.98) is a good sign, as it indicates that the validation set effectively covers the training feature space.

In addition, the **heatmap** visualizes the distribution of data points within the feature space, making it easier to detect gaps and imbalances.

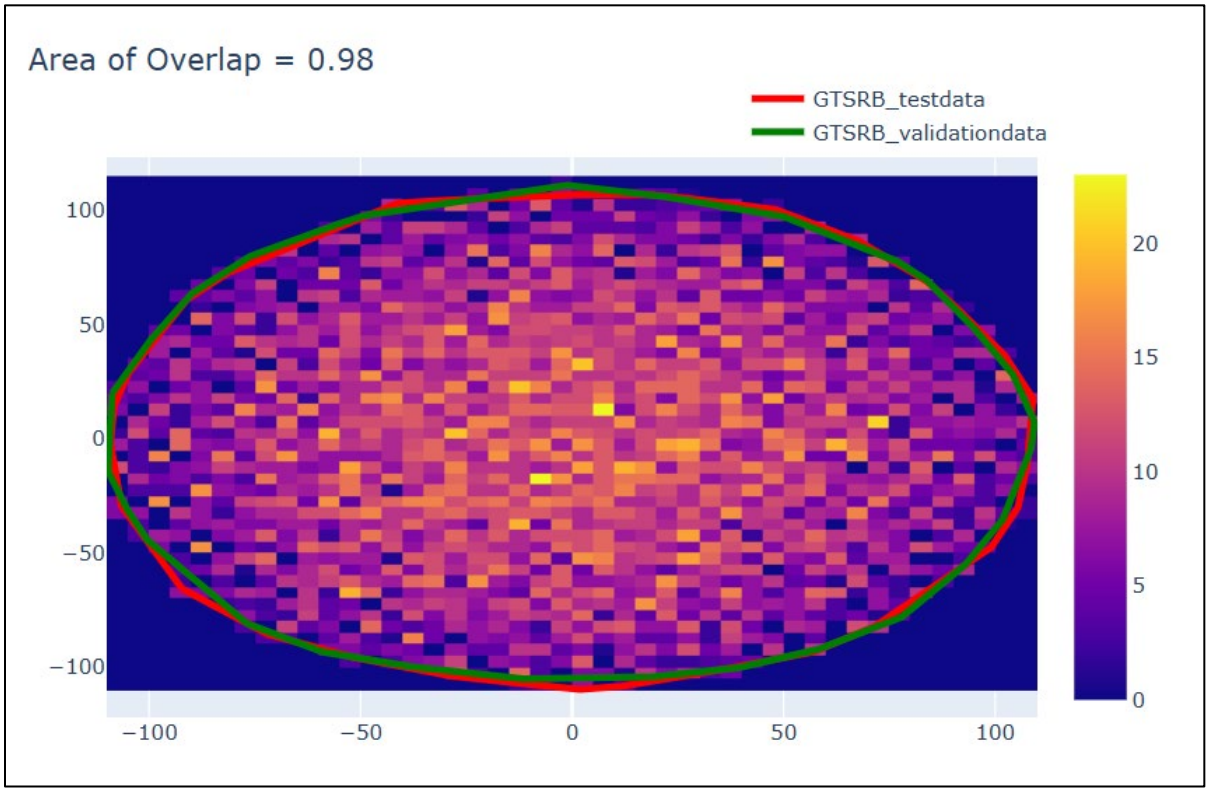


Figure 3. Example of an area of overlap analysis between training and validation datasets.

With these features from stage 2, you ensure that your dataset is well-structured, balanced, and free from hidden dependencies. By continuously validating feature relevance, scaling, and distribution integrity, you **minimize systemic risks and create a solid foundation for robust, interpretable, and compliant model training.**

Stage 3: Model Training

Training is where AI models learn patterns, but also where risks can emerge. Keysight's AI Software Integrity Builder integrates monitoring capabilities during training to track **convergence, detect anomalies, and validate adherence to predefined safety and performance metrics**. This ensures models are not only optimized for accuracy but also aligned with domain-specific constraints and regulatory requirements.

Why it matters:

- Neural networks are universal approximators, but they offer no guarantee of finding an optimal solution - only approximations. Without proper oversight, they may also learn spurious correlations.
- Detecting overfitting and underfitting at an early stage enables stopping training sooner, reducing costs, and improving efficiency. It also helps prevent performance degradation during deployment - a valuable side benefit.
- Continuous monitoring ensures convergence and stability.

To achieve this, our solution provides key capabilities for robust and transparent training:

- **Convergence tracking:** Monitors training progress and stability, helping determine when training should stop – either due to success or failure.
- **Spurious correlation detection:** Identifies misleading patterns that compromise reliability.
- **Over- and underfitting analysis:** Flags imbalance between training and generalization.
- **Safety metric validation:** Confirms compliance with domain-specific constraints and regulatory requirements.
- **Uncertainty quantification:** Uses techniques such as Monte Carlo Dropout to estimate prediction uncertainty and prevent overfitting to the training dataset.
- **Failure diagnostics:** Provides insights into why training failed, enabling cost savings and faster iteration.

Figure 4 below shows an **example of disparate impact, which reveals bias in the data**. Here, for the metadata feature brightness, some buckets are empty - meaning no data points are available - while others show uneven performance. Certain brightness ranges are favored by the model, while others perform worse, suggesting a slight bias.

Detecting such patterns early allows corrective actions before deployment, improving fairness and compliance.

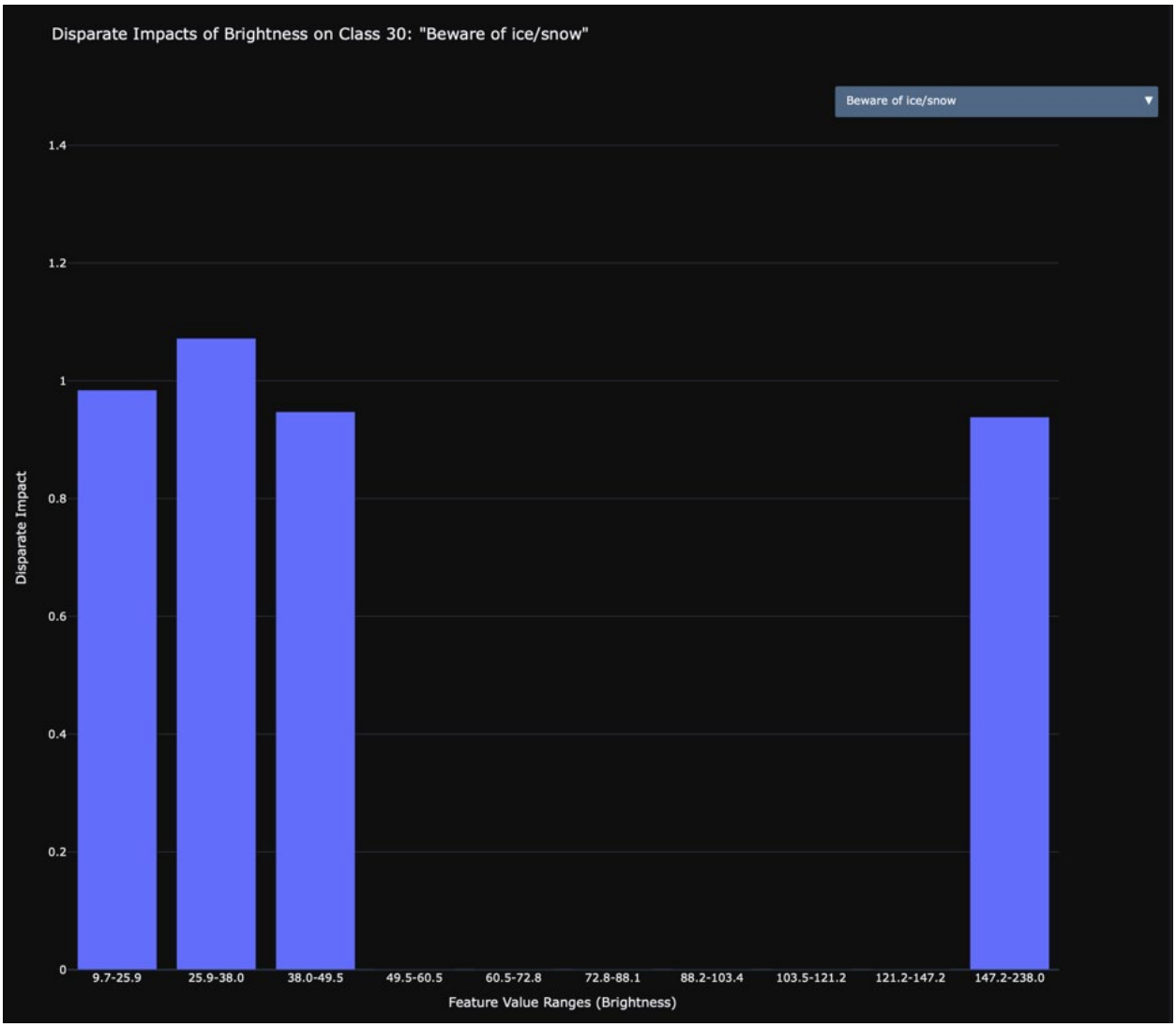


Figure 4. Exemplary bias analysis during training: Disparate impacts

With these training assurance methods from stage 3, you ensure that your **model converges reliably, avoids shortcuts and spurious correlations, and meets both performance and safety criteria -** laying the groundwork for trustworthy model evaluation.

Stage 4: Model Evaluation

Beyond standard accuracy metrics, evaluation must address robustness, fairness, and explainability. Keysight's AI Software Integrity Builder enables **multi-dimensional model assessment**, helping developers answer critical questions such as:

Is the model safe to use? Is it ready for deployment? Did it generalize? Has it met all requirements?

This stage provides the **basis for proving the safety evidence required by regulatory frameworks** such as ISO/PAS 8800 and ensures that AI systems are not only performant but also trustworthy.

Why it matters:

- Accuracy alone does not guarantee safety or compliance
- Robustness and fairness are essential for real-world reliability
- Explainability supports auditability and regulatory conformance

To achieve this, our solution provides comprehensive evaluation capabilities.

Adversarial Robustness Testing

Robustness testing checks how well a model withstands adversarial perturbations - small, calculated changes to inputs that can cause incorrect predictions.

- **Fast Gradient Sign Method (FGSM):** Single-step perturbation for quick robustness checks. Detects if the model fails under slight input changes.
- **Targeted FGSM:** Forces misclassification into a specific target class to reveal decision boundary weaknesses.
- **Iterative FGSM:** Applies multiple small steps for stronger and more realistic adversarial examples, providing deeper robustness evaluation.

Figure 5 illustrates **how the model's predictions change with different values of epsilon**, which represent the strength of adversarial perturbations. As epsilon increases, the input is slightly but systematically altered to test the model's resilience.

Large shifts in predictions at low epsilon values indicate weak robustness, while **stability across a range of values suggests the model can withstand adversarial attacks** without compromising reliability.

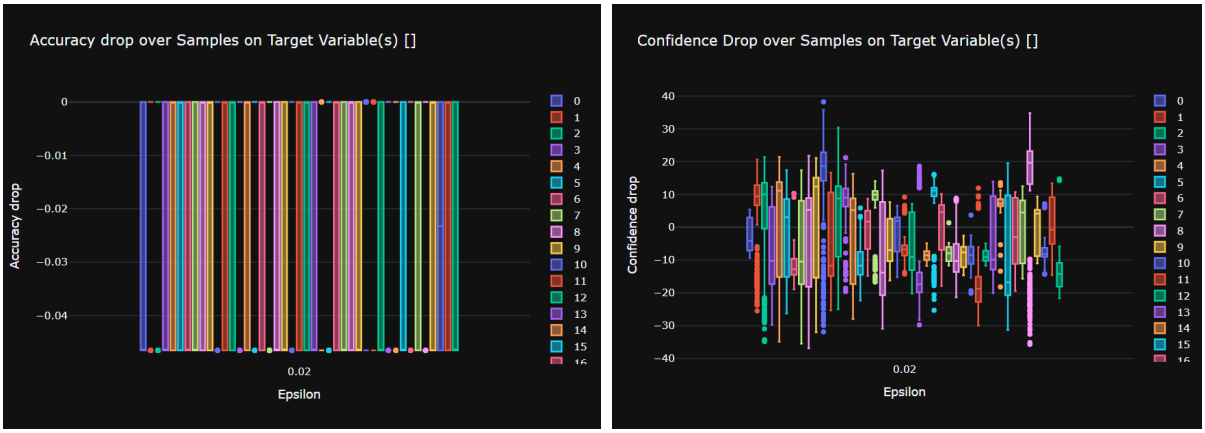


Figure 5. Example of an Adversarial Robustness Analysis

Explainability Tools

After testing robustness, the **next step is to understand why a model made a specific prediction**. Robustness checks reveal whether a model withstands adversarial perturbations, but they do not explain the underlying decision logic. Explainability bridges this gap, **providing transparency into how predictions are made** and helping developers and auditors interpret corner cases and hidden patterns.

- LIME and SHAP*:* Provides local interpretability by explaining why the model made specific predictions. This is critical for understanding corner cases and detecting hidden patterns when combined with other metrics.

Uncertainty Estimation

Models should not only predict but also indicate confidence in their predictions. Keysight's AI Software Integrity Builder uses Monte Carlo Dropout to estimate uncertainty by running multiple stochastic forward passes with dropout enabled during inference.

- **Monte Carlo Dropout:** Runs multiple stochastic forward passes with random dropout layers to quantify prediction confidence and estimate uncertainty.
- **Identifies classes with high uncertainty**, enabling targeted improvements and reducing systemic risk. Low-confidence outputs may require human review or retraining.

Figure 6 shows an example of **boxplots representing prediction uncertainty for each label class**. Narrow or invisible boxes indicate low uncertainty, while visible boxes indicate higher uncertainty. Blue dots represent outliers, and red lines show the average confidence without dropout**. Classes with wider uncertainty ranges may require additional data or model refinement.**

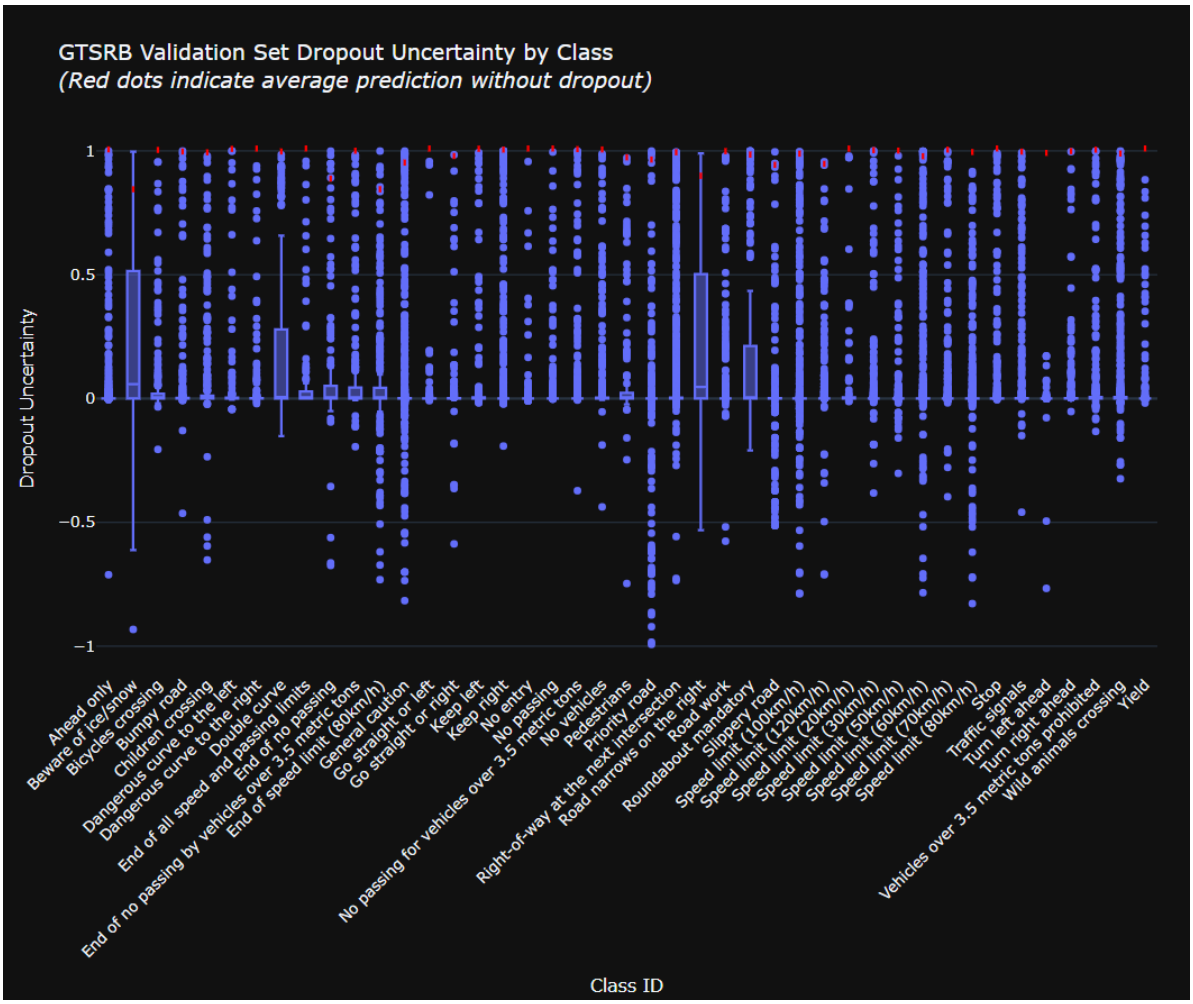


Figure 6. Example of a Monte Carlo Dropout uncertainty analysis

Generalization and Requirement Validation

After uncertainty estimation, the model evaluation continues with generalization validation, which detects blind spots and poor performance on test data. Then, requirement validation confirms compliance with regulatory frameworks and project-specific safety benchmarks.

- **Generalization Validation**: Detects blind spots and poor performance on test data, ensuring the model can handle edge cases and unseen scenarios.
- **Requirement Validation**: Confirms adherence to regulatory frameworks (e.g. ISO/PAS 8800, EU AI Act) and project-specific requirements. Provides documented evidence for audits and certification processes.

By combining robustness, interpretability, uncertainty, and compliance checks, you create a comprehensive **assurance layer that validates the model's readiness for deployment** and builds trust with regulators and end-users.

Stage 5: Continuous Inferencing

Real-world deployment introduces dynamic conditions that static validation cannot capture. Keysight's AI Software Integrity Builder includes **inference-based testing** to monitor data drift, performance degradation, and unexpected behaviors during operation. This **continuous feedback loop supports iterative improvement and lifecycle compliance**, ensuring that AI systems remain safe and reliable over time.

Why it matters:

- Constant change bears the danger of progressive failures.
- Model shift and data drift can compromise safety and accuracy.
- Continuous monitoring is essential to detect failures early and find the optimal point for retraining.

To achieve this, our solution provides advanced monitoring capabilities for continuous assurance.

Data Drift Detection

Data drift occurs when the statistical properties of input data change over time compared to the training set. Even small shifts can lead to inaccurate predictions and compromise safety-critical decisions. Continuous drift detection ensures models stay aligned with real-world conditions and prevents failures. Early drift detection supports proactive retraining, maintains compliance with lifecycle safety requirements, and reduces operational risk.

- **Autoencoder-Based Drift Detection:** Trains an autoencoder on the original training data. During inference, reconstruction error indicates how far new data deviates from the learned distribution - higher errors signal that new data is out of distribution.
- **Baseline Coverage Check for Reliable Drift Interpretation**
  Drift detection is only meaningful if the validation dataset adequately represents the training feature space. Without this baseline coverage, drift metrics can be misleading—what appears as drift might simply reflect gaps in validation coverage.
  - The **Area Overlap Analysis** quantifies how well validation data covers the training feature space. In Stage 2, this analysis ensures validation quality by reducing blind spots and improving generalization. In Stage 5, its role shifts: it provides essential context for interpreting drift signals during deployment.

Figure 7 shows the overlap in the feature space between the training and validation datasets. The red convex hull represents the training data, while the green convex hull represents the validation data. The heatmap indicates data density, with warmer colors indicating higher point concentrations. The Area of Overlap score (here, 0.38) quantifies the extent of overlap between the two hulls. **A low score signals significant coverage gaps, meaning parts of the training space were never validated. These blind spots increase sensitivity to data drift and raise the risk of failures during deployment.** Detecting such gaps early enables better interpretation of drift signals and supports targeted retraining strategies.

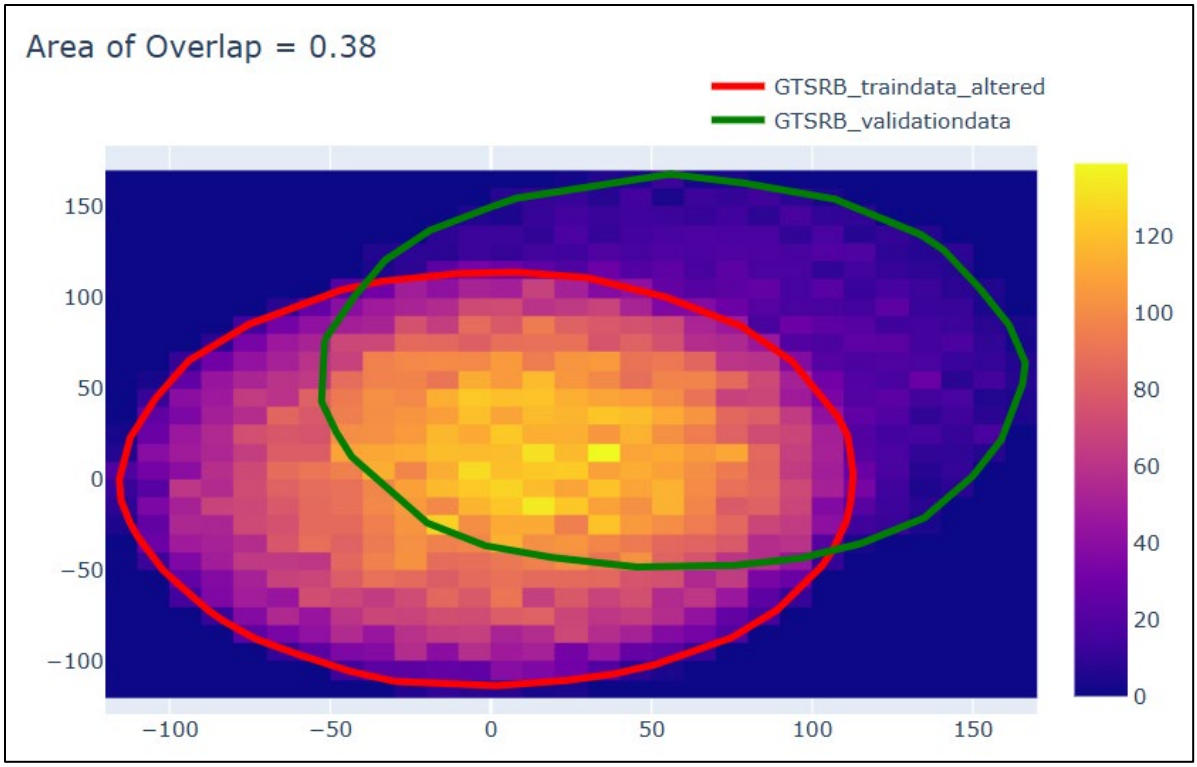


Figure 7. Area of overlap analysis for continuous inferencing analysis

Failure Prediction

Failure prediction uses **anomaly detection and statistical trend analysis to flag unusual patterns** before they lead to operational failures. For example, sudden shifts in prediction confidence or unexpected spikes in error rates can indicate emerging risks. This proactive approach reduces downtime, prevents cascading failures, and ensures that corrective actions are taken before safety or performance is compromised.

Performance Monitoring

Performance monitoring goes beyond simple accuracy checks. It continuously tracks key metrics - such as precision, recall, latency, and confidence scores - against baseline values established during validation. This enables **early detection of gradual degradation**, which can compromise model decisions over time. By identifying trends rather than waiting for failures, you can proactively maintain reliability and compliance.

Retraining Optimization

Retraining is costly and disruptive if done too often - or too late. Keysight's solution analyzes **drift severity**, **performance trends**, and **operational requirements** to determine the **optimal retraining point**. This ensures models remain compliant and reliable without unnecessary cycles, balancing cost efficiency with safety. By aligning retraining decisions with real-world conditions, you avoid both over-maintenance and under-maintenance risks.

Figure 8 shows the distribution of **Out-of-Distribution (OOD) scores**, which measure how far incoming data deviates from the training distribution. **The smaller the score, the more in-distribution the sample is; a score of 0 indicates complete out-of-distribution**. High OOD scores signal that the model is encountering unfamiliar conditions, increasing the risk of inaccurate predictions.

In the plot, well-clustered low scores suggest stable conditions, while wide or skewed distributions indicate significant data drift. The current distribution shows signs of drift, suggesting that a new training cycle should be considered. Detecting these patterns early enables targeted retraining and prevents progressive failures in deployment.

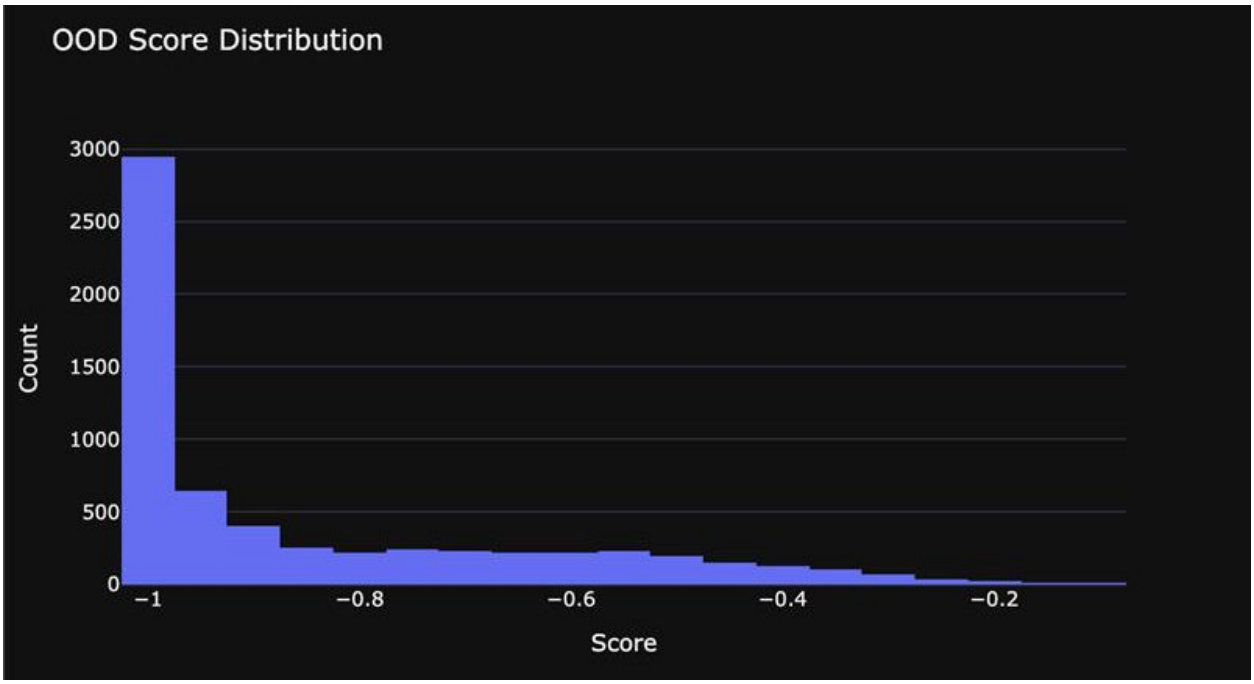


Figure 8. Exemplary Out-of-Distribution (OOD) Score Analysis

By implementing continuous monitoring and feedback loops, you safeguard long-term performance, detect failures early, and optimize retraining - ensuring your AI system remains safe, reliable, and compliant throughout its lifecycle.

Customize as Needed

The analytics and metrics provided by the AI Software Integrity Builder are adaptable to your specific requirements. They depend on factors such as the intended use case, dataset characteristics, and model architecture. This flexibility ensures that validation aligns with real-world conditions and your operational goals.

While customization is key, the solution provides a strong foundation of essential validation methods and metrics, ready to deliver immediate insights and accelerate the AI assurance process. As your projects evolve, these capabilities can be extended and tailored to your specific needs, **creating a scalable and future-proof AI validation strategy**.

Contact us for more information on your specific customization needs.

Collaborative AI Validation with the AI Software Integrity Builder

Keysight's AI Software Integrity Builder is designed to enable seamless collaboration across all stakeholders in the AI development process: data scientists, AI engineers, domain experts, quality assurance teams, and safety officers.

One of its core principles is **bridging the gap between domain expertise and AI engineering**. Domain experts can define relevant test scenarios and metrics, ensuring that validation aligns with real-world requirements. This collaborative approach enhances model quality and reliability, ultimately leading to safer and more effective AI systems.

Function, System and Domain Validation for Trustworthy AI Deployment

AI failures often occur due to system-level interactions or domain-level gaps, not just model errors. Holistic validation goes beyond the AI model itself. It must consider the model's integration into the overall system and the adaptation to the operational design domain (ODD).

Keysight's AI Software Integrity Builder provides a **unified AI assurance approach** with metrics and tools to evaluate AI at the function, system, and domain level, ensuring they perform reliably and safely within complex and safety-critical environments.

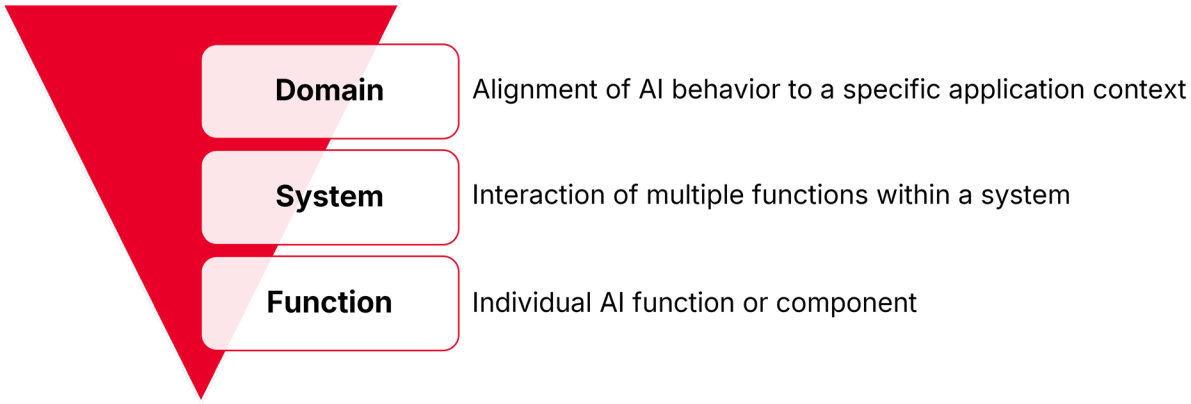


Figure 8. Holistic AI assurance involves the function, system and domain level.

By validating AI at all levels, Keysight delivers a **comprehensive AI assurance framework throughout development and maintenance,** ultimately enabling safer, more effective AI-enabled solutions and trustworthy AI deployment in conformance with regulatory requirements.

Ensure compliance, reliability, and trustworthiness in every step of your AI lifecycle. Contact Keysight to build a future-proof AI assurance approach for your organization.