INTERNATIONAL JOURNAL OF DATA SCIENCE AND MACHINE LEARNING (ISSN: 2692-5141)
VOL.06 ISSUE 02 (2026)

Research Article

Explainable Machine Learning Framework for Predicting Solid-State Drive Failures Using LIME and SHAP

Kwame A. Mensah[1]

[1]Department of Artificial Intelligence, Accra Institute of Technology, Accra, Ghana

Received: 12 Jun 2026
Revised: 2 Jul 2026
Accepted: 20 Aug 2026
Published: 04 Sep 2026
Doi:10.55640/ijdsml-06-02-08
Page No: 169-177

Abstract

The Solid-State Drives (SSDs) have become critical components of modern computing infrastructures because of their high throughput, low access latency, and increasing storage density. However, SSD failure can cause data unavailability, service interruption, and operational costs, making reliable failure prediction an important research problem. Machine learning provides a promising approach for identifying complex relationships between operational characteristics and impending device failures, but the opacity of predictive models can limit their practical adoption. This paper proposes an explainable machine learning framework for SSD failure prediction that integrates predictive modeling with Local Interpretable Model-Agnostic Explanations (LIME) and SHapley Additive exPlanations (SHAP). The framework is designed around data preparation, health-indicator construction, feature selection, predictive modeling, model validation, local explanation, global feature attribution, and explanation consistency analysis. The supplied literature demonstrates growing interest in combining advanced computational intelligence with complex predictive and optimization tasks, while the directly relevant reference emphasizes transparency in SSD failure prediction (Kumar, 2026). The proposed framework extends this direction by treating explainability as an integral component of the prediction pipeline rather than an after-the-fact visualization mechanism. Analytical findings indicate that combining predictive accuracy with local and global explanations can improve interpretability, support failure investigation, and provide a more actionable basis for maintenance decisions. The study also identifies limitations associated with explanation stability, correlated variables, temporal degradation patterns, and the domain relevance of the currently supplied literature. The resulting framework establishes a research-oriented architecture for trustworthy and operationally interpretable SSD failure prediction.

Keywords: Solid-State Drive Failure Prediction, Explainable Artificial Intelligence, Machine Learning, LIME, SHAP, Predictive Maintenance, Storage Reliability, Feature Importance, Model Interpretability

INTRODUCTION

The increasing dependence on digital storage infrastructure has made storage-device reliability an important component of system availability. SSDs differ fundamentally from conventional magnetic storage devices because their operation depends on flash-memory cells, controllers, wear-management mechanisms, error-correction processes, and workload-dependent access patterns. Consequently, failure may emerge through complex interactions among multiple health indicators rather than through a single observable variable. A predictive system must therefore identify patterns associated with

degradation while distinguishing normal operational variation from meaningful indications of impending failure.

Machine learning is particularly suitable for this problem because failure behavior may be nonlinear, multidimensional, and difficult to represent through manually defined thresholds. A predictive model can learn relationships between historical device observations and failure outcomes, potentially enabling proactive maintenance. Nevertheless, predictive capability alone is insufficient for safety- and reliability-sensitive infrastructure. A model that predicts a device as high risk without explaining why that prediction was produced may be difficult for storage administrators or reliability engineers to trust.

This concern motivates the integration of Explainable Artificial Intelligence (XAI) into SSD failure prediction. Kumar (2026) specifically positions LIME and SHAP as mechanisms for improving transparency in SSD failure prediction. In this context, explainability is not merely a presentation feature; it represents a mechanism for translating statistical model behavior into evidence that can be examined by technical decision-makers.

The broader supplied literature also demonstrates an increasing movement toward computational systems capable of handling complex optimization, generation, and decision processes. Evolutionary computation has been integrated with large language models for code completion, optimization code generation, recommender systems, neural architecture search, and other computational tasks (Liu et al., 2026; Ma et al., 2026; Lai et al., 2026; Li et al., 2026). These studies collectively suggest that modern intelligent systems increasingly combine automated learning with mechanisms for improving computational effectiveness. However, their direct relevance to SSD reliability is limited. This distinction is important because a literature synthesis should not imply that techniques developed for LLM optimization or code generation automatically establish evidence for storage failure prediction.

Accordingly, this study has four principal objectives. First, it develops a structured machine-learning architecture for SSD failure prediction. Second, it integrates LIME and SHAP into the prediction pipeline to provide complementary local and global explanations. Third, it establishes an analytical procedure for evaluating predictive outputs and explanation quality simultaneously. Fourth, it identifies methodological limitations and future research requirements for trustworthy SSD prognostics.

The significance of the proposed framework lies in its integration of prediction and interpretation. Instead of asking only whether an SSD is likely to fail, the framework asks which observed characteristics contributed to the prediction, whether those contributions are stable, and whether the explanation can support an appropriate maintenance decision.

LITERATURE REVIEW

2.1 Computational Intelligence and Explainable Prediction

The supplied literature predominantly examines evolutionary computation and its interaction with large language models. Lai et al. (2026) investigate evolutionary neural architecture search guided by large language models, illustrating how intelligent search mechanisms can be integrated into model-development processes. Similarly, Ma et al. (2026) examine instruction tuning for optimization code generation, while Liu et al. (2026) investigate evolutionary computation-enhanced large language models for intelligent code completion. These studies demonstrate the broader movement toward adaptive computational frameworks in which automated methods are used to improve complex machine-learning workflows.

Li et al. (2026) examine language-model evolutionary algorithms for recommender

systems and emphasize benchmarking and algorithm comparison. Their work is relevant conceptually because it highlights the importance of evaluating intelligent systems beyond a single performance criterion. For SSD failure prediction, the same principle suggests that predictive accuracy should not be considered independently from interpretability, robustness, and operational usefulness.

Zhang et al. (2026) explore a lightweight large language model driven by evolutionary computation. The emphasis on lightweight computational architectures is relevant to SSD monitoring because predictive systems deployed in operational environments may need to operate under resource constraints. However, the reference does not provide direct evidence about storage-device prognostics and therefore cannot be interpreted as SSD-specific validation.

2.2 Optimization, Adaptation, and Complex Decision Systems

The supplied studies also cover computational optimization in diverse contexts. Li et al. (2026) investigate a co-evolutionary approach involving programs and test cases, whereas Li et al. (2026) propose an LLM-assisted memetic algorithm for hybrid job-shop scheduling. Hou et al. (2026) examine evolutionary content generation using multimodal LLM-based fitness evaluation. Collectively, these studies reinforce the importance of adaptive evaluation mechanisms in complex computational systems.

Han et al. (2026) investigate multigranularity adversarial attacks on large language models using genetic programming, while You et al. (2026) examine an evolutionary token-level jailbreaking framework. Although these studies concern security and robustness of language models rather than storage devices, they reveal a broader methodological concern: intelligent systems may exhibit complex behaviors that cannot be fully characterized through a single aggregate performance measure.

This observation has methodological relevance for SSD prediction. A failure-prediction model may achieve strong classification performance while still producing unstable or misleading explanations. Consequently, the proposed framework treats explanation assessment as a complementary analytical dimension.

2.3 Directly Relevant SSD Explain ability Research

Among the supplied references, Kumar (2026) is the most directly aligned with the present research problem. The work explicitly addresses SSD failure prediction using LIME and SHAP for transparency. This establishes the conceptual basis for integrating model-agnostic explainability into storage failure analysis.

LIME provides local interpretability by approximating the behavior of a complex predictive model around an individual observation. Its principal value in SSD monitoring is that it can answer a device-specific question: why did the model assign this particular drive a high or low failure probability? SHAP provides a complementary attribution perspective by estimating the contribution of individual features to a model output. This allows analysts to examine both individual predictions and broader feature influence patterns (Kumar, 2026).

The central research opportunity is therefore not simply to introduce LIME and SHAP independently, but to design a unified framework in which their outputs can be compared, validated, and translated into maintenance-oriented information.

2.4 Research Gap

The supplied references reveal a substantial domain-specific gap. Most references focus on evolutionary computation, LLMs, optimization, recommender systems, code generation, scheduling, and adversarial behavior (Han et al., 2026; Hou et al., 2026; Lai et al., 2026; Li et al., 2026; Liu et al., 2026; Ma et al., 2026; Zhang et al., 2026; You et al.,

2026). They do not provide a broad empirical foundation for SSD failure prediction.

Therefore, this study does not treat those works as direct evidence of SSD predictive performance. Instead, they provide contextual support for the broader methodological movement toward adaptive, intelligent, and interpretable computational systems. The direct SSD research foundation is represented by Kumar (2026), from which the proposed framework develops a more explicit end-to-end architecture.

METHODOLOGY

3.1 Proposed Framework Architecture

The proposed Explainable SSD Failure Prediction Framework (ESFPF) consists of seven interconnected stages:

SSD operational data → preprocessing → feature engineering → predictive modeling → performance evaluation → LIME/SHAP explanation → maintenance-oriented interpretation.

The architecture is designed to preserve a separation between prediction and explanation. The predictive model first produces a failure-risk estimate. Explanation modules then analyze the model's decision process. This separation prevents the interpretability layer from being confused with the underlying prediction mechanism.

The framework can accept device-level observations such as operational health indicators, error-related measurements, workload characteristics, utilization information, and historical status observations. The exact feature set should depend on the SSD monitoring dataset used in an implementation.

3.2 Data Preprocessing

SSD telemetry may contain missing values, inconsistent sampling intervals, noisy measurements, and variables with different numerical scales. Preprocessing therefore constitutes an essential stage rather than a purely technical preliminary step.

Missing observations should be examined according to their origin. Randomly absent measurements may be treated differently from systematically missing values caused by device-state transitions. Similarly, extreme observations should not automatically be removed because abnormal values may themselves represent useful failure signals.

For temporal SSD observations, chronological ordering is especially important. Randomly mixing observations from the same device across training and testing datasets may introduce information leakage. A robust implementation should therefore maintain temporal or device-level separation when evaluating predictive performance.

3.3 Feature Engineering and Health Indicators

Raw telemetry may not always provide the most informative representation of degradation. Feature engineering can transform operational observations into indicators representing cumulative wear, error progression, workload intensity, or changes in device behavior.

A particularly important concept is temporal change. Let a health variable for device ii at time tt be represented by xi,tx_{i,t}. A simple change indicator can be expressed as

Δxi,t=xi,t−xi,t−1.\Delta x_{i,t}=x_{i,t}-x_{i,t-1}.

A rolling statistic can similarly be represented as

x¯i,t(w)=1w∑k=0w−1xi,t−k,\bar{x}_{i,t}^{(w)}=\frac{1}{w}\sum_{k=0}^{w-1}x_{i,t-k},

where ww represents the selected temporal window.

These transformations allow the model to represent both the current state and the trajectory of the device. This distinction is important because two SSDs with similar current health values may have different future risks if one is stable while the other is deteriorating rapidly.

3.4 Predictive Modeling Layer

The predictive layer receives the engineered feature vector

X=(x1,x2,…,xp)X=(x_1,x_2,\ldots,x_p)

and produces a failure-risk estimate

y^=f(X).\hat{y}=f(X).

For binary failure prediction, the output may be interpreted as the probability of failure within a predefined prediction horizon. The selected algorithm may be any appropriate supervised learning model capable of handling the characteristics of the available data.

Model selection should not be based solely on accuracy. In imbalanced failure datasets, accuracy can be misleading because a model can obtain a high accuracy value by predicting the majority non-failure class. Precision, recall, F1-score, area under the ROC curve, and, where appropriate, precision-recall analysis should therefore be considered collectively.

3.5 LIME-Based Local Explanation

LIME is incorporated to explain individual SSD predictions. For an observation xx, LIME constructs an interpretable local approximation of the complex model:

gx(z)≈f(z),g_x(z)\approx f(z),

where gxg_x represents an interpretable surrogate around the selected observation.

The practical interpretation is straightforward. Suppose an SSD receives a high predicted failure probability. LIME can identify the features that locally increased or decreased that prediction. A storage administrator can consequently examine whether the model's reasoning corresponds to meaningful device behavior.

The principal advantage of LIME is local specificity. Its limitation is that explanations can vary according to the neighborhood and sampling procedure used to approximate the model. Thus, LIME explanations should not automatically be interpreted as universal statements about feature importance.

3.6 SHAP-Based Global and Local Attribution

SHAP provides a complementary explanation mechanism based on Shapley-value attribution. For a model prediction f(x)f(x), the explanation can conceptually be represented as

f(x)=ϕ0+∑j=1pϕj,f(x)=\phi_0+\sum_{j=1}^{p}\phi_j,

where ϕj\phi_j represents the contribution attributed to feature jj and ϕ0\phi_0 represents the baseline output.

The framework uses SHAP at two levels. At the local level, SHAP identifies which features

contributed to an individual SSD's predicted risk. At the global level, aggregation of absolute SHAP contributions can identify variables that exert substantial influence across the dataset.

The distinction between LIME and SHAP is therefore useful. LIME emphasizes local approximation, whereas SHAP provides a theoretically grounded feature-attribution representation. Using both creates an opportunity for cross-validation of explanations rather than relying on one interpretability mechanism (Kumar, 2026).

3.7 Explanation Consistency Analysis

A major methodological extension of the framework is explanation consistency. If LIME indicates that feature x1x_1 strongly increases risk while SHAP indicates a substantially different attribution, the discrepancy should be investigated rather than ignored.

For a selected observation, the framework compares the ranked feature contributions generated by both explainers. A simple conceptual agreement measure can be represented as

C(x)=|RL(x)∩RS(x)||RL(x)∪RS(x)|,C(x)=\frac{|R_L(x)\cap R_S(x)|}{|R_L(x)\cup R_S(x)|},

where RLR_L and RSR_S denote the selected important-feature sets produced by LIME and SHAP.

This measure is not proposed as a universal explanation metric; rather, it provides an analytical mechanism for identifying agreement and disagreement between explanation methods.

3.8 Decision-Oriented Interpretation

The final framework stage translates explanations into operational categories. A prediction may indicate low, moderate, or high risk, while the explanation identifies the principal contributing factors.

For example, if an SSD is classified as high risk and multiple independent indicators consistently contribute to the prediction, the case may receive higher maintenance priority. Conversely, if a high-risk prediction is driven by a single unstable feature and explanation methods disagree substantially, the case may require additional validation before replacement.

This approach prevents explainability from becoming merely a graphical component. Its purpose is to connect model output with a technically defensible decision process.

RESULTS

The analytical evaluation of the proposed framework produces several principal findings. First, integrating explainability directly into the SSD prediction pipeline changes the interpretation of machine-learning output from a binary or probabilistic classification into an evidence-supported assessment. A failure-risk prediction becomes more operationally useful when accompanied by the variables that contributed to that prediction. This principle is consistent with the transparency-oriented direction of Kumar (2026).

Second, the framework establishes a complementary relationship between local and global interpretation. LIME can identify the reasons associated with an individual SSD, whereas SHAP can be used to examine feature contributions at both individual and aggregated levels. This distinction is particularly important for predictive maintenance because maintenance decisions are inherently device-specific, while model validation requires population-level understanding.

Third, the framework indicates that feature importance should not be interpreted as a direct causal relationship. A feature receiving a high SHAP contribution means that the model relied substantially on that variable, not that the variable independently caused the SSD failure. This distinction protects the framework from over-interpreting machine-learning associations.

Fourth, explanation agreement provides an additional quality-control mechanism. When LIME and SHAP identify similar dominant variables for a prediction, confidence in the interpretive consistency of the result can increase. Conversely, substantial disagreement indicates a case requiring additional investigation. Therefore, explanation disagreement should be treated as diagnostic information rather than automatically interpreted as model failure.

Fifth, temporal feature engineering is analytically important. A static measurement can describe the current state of an SSD, whereas a change rate or rolling statistic can describe its trajectory. The proposed framework consequently supports a more comprehensive representation in which both state and degradation dynamics can influence prediction.

Sixth, the literature analysis demonstrates that the current reference base is insufficient for claiming universal empirical superiority of the proposed framework. The majority of supplied references address evolutionary computation and LLM-related problems rather than SSD reliability. Consequently, the findings support a methodological framework rather than a statistically validated claim that LIME-SHAP integration necessarily outperforms every alternative SSD prediction architecture.

Finally, the framework highlights an important distinction between predictive effectiveness and operational trust. A model can potentially classify failures effectively while providing unstable explanations. Conversely, a highly interpretable model may lack sufficient predictive sensitivity. The proposed architecture therefore treats predictive performance and explanatory quality as complementary evaluation dimensions.

DISCUSSION

The central contribution of the proposed framework is the integration of prediction, explanation, and decision interpretation into a single SSD reliability architecture. Traditional predictive workflows often terminate when the model generates a probability or class label. Such an approach is insufficient when engineers need to understand why a device has been classified as high risk. The integration of LIME and SHAP addresses this interpretability problem by creating both local and broader attribution perspectives (Kumar, 2026).

The theoretical implication is that explainability should be considered part of model evaluation rather than merely part of model presentation. LIME and SHAP provide different explanatory mechanisms, and their combination allows the analyst to examine whether important features remain consistent across explanation strategies. This is especially valuable when SSD telemetry contains correlated variables. If several health indicators describe related aspects of the same underlying degradation process, feature attribution can become difficult to interpret independently.

A second implication concerns the distinction between prediction and causation. SHAP or LIME may identify a variable as influential without establishing that manipulating the variable would prevent failure. Consequently, maintenance policies should not be constructed directly from feature rankings without engineering validation. Explanations should instead serve as diagnostic evidence that guides further investigation.

The framework also exposes a trade-off between model complexity and interpretability. More sophisticated models may capture nonlinear interactions and complex degradation relationships, but their internal decision mechanisms can be difficult to understand.

Model-agnostic explainers address part of this challenge, but they do not eliminate it. Explanation quality depends on the fidelity and stability of the approximation or attribution process.

The broader supplied literature reinforces the importance of this issue from a different direction. Research involving evolutionary computation and LLMs demonstrates the increasing complexity of modern intelligent systems and the need for systematic evaluation (Lai et al., 2026; Li et al., 2026; Liu et al., 2026). However, these studies should not be interpreted as direct evidence for SSD failure prediction. Their relevance is methodological: they illustrate the broader trend toward increasingly sophisticated computational systems whose behavior requires structured evaluation.

A significant limitation is therefore the current evidence base. The provided references contain only one directly SSD-focused study, while the remaining references concern other application domains. Consequently, claims regarding comparative predictive performance, generalization across SSD manufacturers, or superiority over established prognostic techniques cannot responsibly be established from the supplied literature alone.

Another limitation concerns temporal dependence. LIME and SHAP explanations are generally associated with individual model inputs, while SSD degradation is fundamentally sequential. A feature may appear important at one point because of its historical trajectory rather than its instantaneous value. Future implementations should therefore investigate explanation mechanisms capable of explicitly representing temporal dependencies.

Finally, explanation stability must be evaluated under changes in sampling, feature correlation, model configuration, and observation windows. An explanation that changes substantially under minor perturbations may have limited operational value. Thus, trustworthy SSD prediction requires not only accurate classification but also stable, coherent, and technically interpretable explanations.

CONCLUSION

This study presented an Explainable Machine Learning Framework for Predicting Solid-State Drive Failures Using LIME and SHAP. The proposed architecture integrates data preprocessing, temporal feature engineering, predictive modeling, performance assessment, local explanation, global attribution, explanation consistency analysis, and maintenance-oriented interpretation.

The framework's principal contribution is its treatment of explainability as an integral component of SSD failure prediction rather than a secondary visualization layer. LIME provides localized insight into individual predictions, while SHAP supports systematic feature attribution across both individual observations and the broader dataset. Their complementary use can help transform opaque failure-risk predictions into interpretable evidence suitable for technical investigation.

The analysis also demonstrates that explanation should not be equated with causality. Feature contributions identify how a model arrived at a prediction, not necessarily which physical mechanism caused device failure. This distinction is essential for responsible deployment in reliability-sensitive environments.

The supplied literature further reveals an important research gap. Although recent work demonstrates sophisticated computational intelligence across evolutionary algorithms, LLMs, optimization, code generation, recommender systems, and adversarial analysis, these studies do not provide extensive direct evidence for SSD failure prediction. The present framework therefore establishes a focused methodological foundation rather than claiming unsupported empirical superiority.

Future research should evaluate the framework using longitudinal SSD telemetry containing both healthy and failed-device histories. Particular attention should be given to temporal cross-validation, class imbalance, explanation stability, correlated features, cross-device generalization, and agreement between LIME and SHAP. Comparative experiments should also evaluate whether explanation-aware modeling improves maintenance decisions without sacrificing failure-detection sensitivity.

REFERENCES

1. W. Han, H. Li, M. Gong, Y. Zhou, Y. Wu, and A. K. Qin, "Multigranularity adversarial attacks on large language models using genetic programming," IEEE Trans. Evol. Comput., vol. 30, no. 4, pp. 1465– 1479, Aug. 2026.

2. Y. Hou, N. Wang, Z. Yu, D. M. Bossens, Y. Wu, and Q. Zhang, "Evolutionary content generation via multimodal llm-based fitness evaluation," IEEE Trans. Evol. Comput., vol. 30, no. 4, pp. 1435– 1449, Aug. 2026.

3. Y. Lai, Z. Cai, L. Chen, T. Ling, and H.-L. Liu, "LLMENAS: Evolutionary neural architecture search via large language model guidance," IEEE Trans. Evol. Comput., vol. 30, no. 4, pp. 1362–1376, Aug. 2026.

4. K. Li, Y. Yuan, H. Yu, T. Guo, and S. Cao, "CoCoEvo: Co-evolution of programs and test cases to enhance code generation," IEEE Trans. Evol. Comput., vol. 30, no. 4, pp. 1420–1434, Aug. 2026.

5. R. Li, L. Wang, H. Sang, L. Yao, and L. Pan, "LLM-assisted automatic memetic algorithm for lot-streaming hybrid job shop scheduling with variable sublots," IEEE Trans. Evol. Comput., vol. 30, no. 4, pp. 1377– 1389, Aug. 2026.

6. J. Li, Z. Sun, S. Feng, C. Chen, and Y.-S. Ong, "Language model evolutionary algorithms for recommender systems: benchmarks and algorithm comparisons," IEEE Trans. Evol. Comput., vol. 30, no. 4, pp. 1405–1419, Aug. 2026.

7. D. Liu, Z. Tan, G. G. Yen, S. Duan, Y. Zhou, and Z. He, "Evolutionary computation-enhanced large language models for intelligent code completion," IEEE Trans. Evol. Comput., vol. 30, no. 4, pp. 1347– 1361, Aug. 2026.

8. Z. Ma, Y.-J. Gong, H. Guo, J. Chen, Y. Ma, and Z. Cao, "LLaMoCo: Instruction tuning of large language models for optimization code generation," IEEE Trans. Evol. Comput., vol. 30, no. 4, pp. 1390– 1404, Aug. 2026.

9. M. Zhang, W. Wei, Z. Zhou, W. Liu, J. Zhang, and A. Belatreche, "Spike-driven lightweight large language model with evolutionary computation," IEEE Trans. Evol. Comput., vol. 30, no. 4, pp. 1333– 1346, Aug. 2026.

10. S. You et al., "UniBreak: A unified evolutionary token-level jailbreaking framework for large language models," IEEE Trans. Evol. Comput., vol. 30, no. 4, pp. 1450–1464, Aug. 2026.

11. Kumar, S. K. (2026). Explainable AI for SSD Failure Prediction: Using LIME and SHAP for Transparency. Journal of Engineering Research and Sciences, 5(4), 1–16. https://doi.org/10.55708/js0504001