



**Dissertation Submitted to the Department of 2<sup>nd</sup> cycle training in Partial  
Fulfillment of the Requirements for Engineer's Degree in Computer Science**  
**Specialty: Artificial Intelligence and Data Sciences**

Submitted By:

**KABOUB Younes**

**AIOps-Driven Predictive Maintenance of SSDs in Hyperscale  
Data Centers**

Supervised by:

**Dr. MEDJOUDJ Rafik (ESTIN)**

**Members of jury:**

- |                                |           |       |
|--------------------------------|-----------|-------|
| ▪ <b>Dr. IKEN Sofiane</b>      | President | ESTIN |
| ▪ <b>Dr. Djamaa Kenza</b>      | Examiner  | ESTIN |
| ▪ <b>Ms. Bessaoudi Sabrina</b> | Examiner  | ESTIN |

Academic year: 2025/2026

# Acknowledgements

First and foremost, I thank God for the strength, patience, and clarity of mind that carried this work to completion.

My deepest gratitude goes to my supervisor, **Dr. MEDJOUDJ Rafik**, whose technical rigor, constructive criticism, and constant availability have decisively shaped the direction and quality of this engineering project. His insistence on operational realism over academic comfort is the reason this thesis presents a deployable system rather than a notebook.

I extend my sincere thanks to the faculty of the **École supérieure en Sciences et Technologies de l'Informatique et du Numérique (ESTIN Béjaïa)** for five years of demanding training that built the foundations for this work — particularly the lecturers of the Artificial Intelligence and Data Science specialty who instilled the engineering discipline this project required.

I thank the members of the jury for the time and care they have devoted to evaluating this work, and the Alibaba Cluster Trace Program for making the SSD failure dataset publicly available — without it, neither this thesis nor any reproducible research on hyperscale SSD reliability would be possible.

Finally, my warmest thanks to my family and to my colleagues whose discussions, encouragement, and good humour sustained the long evenings of debugging memory-leaking Pandas pipelines and tracking down silent failures in 4.4 GB of compressed S.M.A.R.T. telemetry.

# Abstract

Hyperscale data centers now operate Solid State Drive (SSD) fleets numbering in the hundreds of thousands to millions of devices. At this scale, the cost of an unplanned SSD failure — measured in service downtime, data migration overhead, and RAID-rebuild I/O amplification — has made the transition from reactive replacement and static S.M.A.R.T. thresholds to AIOps-driven predictive maintenance an operational necessity. This engineering thesis presents **Smart Storage AI**, an end-to-end pipeline for SSD failure prediction validated on the public Alibaba SSD failure dataset (2018–2019), which spans approximately 500,000 SSDs across six drive architectures (MA1, MA2, MB1, MB2, MC1, MC2) and two NAND technologies (MLC and TLC).

The contribution is fourfold. First, a memory-safe **data engineering pipeline** that handles 8.7 GB of compressed daily S.M.A.R.T. telemetry on a 16 GB workstation, performs aggressive ghost-data and silent-failure detection, and produces strictly continuous 40-day pre-failure sequences with a sharp 14-day prediction horizon and a 7-day minimum signal density. Second, a **feature engineering framework** that consolidates five complementary scoring metrics (Pearson, Spearman, Youden’s J-index, Random Forest importance, XGBoost importance) into a single ranking, yielding six dominant attributes ( $n\_5$ ,  $r\_9$ ,  $r\_241$ ,  $r\_242$ ,  $r\_190$ ,  $r\_180$ ) whose causal role is verified through SHAP analysis. Third, a portfolio of **per-architecture champion models** — Random Forest, XGBoost, a causal Attention-BiLSTM, and a soft-voting Supreme Ensemble — trained under strict 5-fold StratifiedGroupKFold cross-validation with disk-level smoothed evaluation. Fourth, a **Universal Blind XGBoost** trained on the full union of all six architectures with no hardware labels, leveraging sparsity-aware tree splits to handle heterogeneous feature sets.

The central empirical finding is that the Universal Blind model achieves near-parity with the per-architecture champions across all six drive models — for the four largest architectures the F1 gap is below 0.01 — while collapsing twenty-four model runs into a single deployable artifact and yielding an estimated reduction of approximately 96% in MLOps overhead (training, monitoring, retraining, and governance). The Attention-BiLSTM champion delivers an average advance warning of 13.30 days on the MB1 architecture at threshold 0.4 with disk-level F1 of 0.7977, while the per-architecture Supreme Ensemble reaches F1 of 0.9338 on the MA1 fleet. The complete system is exposed through Smart Storage AI Dashboard, a Streamlit mission-control dashboard that replays predictions from a 15% blind test vault of 4,453 drives across five interactive pages, with Operator and Engineer modes targeting realistic NOC workflows.

**Keywords:** SSD failure prediction; AIOps; predictive maintenance; hyperscale data centers; XGBoost; Attention-BiLSTM; ensemble learning; sparsity-aware learning; Alibaba SSD dataset; MLOps; dashboard; Streamlit.

# Résumé

Les centres de données hyperscale exploitent désormais des flottes de disques SSD comptant des centaines de milliers à plusieurs millions d'unités. À cette échelle, le coût d'une défaillance SSD non planifiée — mesuré en temps d'indisponibilité de service, surcoûts de migration de données et amplification d'E/S lors de la reconstruction RAID — a rendu la transition d'une maintenance réactive et de seuils S.M.A.R.T. statiques vers une maintenance prédictive pilotée par AIOps une nécessité opérationnelle. Ce mémoire d'ingénieur présente **Smart Storage AI**, un pipeline de bout en bout pour la prédiction des défaillances SSD validé sur le jeu de données public Alibaba SSD (2018–2019), qui couvre environ 500 000 SSD répartis sur six architectures (MA1, MA2, MB1, MB2, MC1, MC2) et deux technologies NAND (MLC et TLC).

La contribution comporte quatre volets. Premièrement, un **pipeline d'ingénierie de données** économe en mémoire, capable de traiter 8,7 Go de télémétrie S.M.A.R.T. quotidienne compressée sur un poste de travail de 16 Go, qui effectue une détection agressive des données fantômes et des défaillances silencieuses, et produit des séquences pré-défaillance de 40 jours strictement continues, avec une fenêtre de prédiction de 14 jours et une densité de signal minimale de 7 jours. Deuxièmement, un **cadre d'ingénierie de caractéristiques** qui consolide cinq métriques de notation complémentaires (Pearson, Spearman, indice de Youden, importance Random Forest, importance XGBoost) en un classement unique, identifiant six attributs dominants ( $n\_5$ ,  $r\_9$ ,  $r\_241$ ,  $r\_242$ ,  $r\_190$ ,  $r\_180$ ) dont le rôle causal est confirmé par une analyse SHAP. Troisièmement, un portefeuille de **modèles champions par architecture** — Random Forest, XGBoost, un Attention-BiLSTM causal et un ensemble Supreme par soft-voting — entraînés sous validation croisée stricte StratifiedGroupKFold à 5 plis avec évaluation lissée au niveau du disque. Quatrièmement, un **XGBoost Universel Aveugle** entraîné sur l'union complète des six architectures sans étiquettes matérielles, qui exploite les divisions d'arbres conscientes de la sparsité pour gérer des ensembles de caractéristiques hétérogènes.

Le résultat empirique central est que le modèle Universel Aveugle atteint une quasi-parité avec les champions par architecture sur les six modèles de disques — pour les quatre plus grandes architectures, l'écart de F1 est inférieur à 0,01 — tout en regroupant vingt-quatre exécutions de modèles en un seul artefact déployable, ce qui produit une réduction estimée à environ 96% des frais MLOps (entraînement, supervision, ré-entraînement et gouvernance). Le champion Attention-BiLSTM fournit un avertissement moyen de 13,30 jours sur l'architecture MB1 au seuil 0,4 avec un F1 disque-niveau de 0,7977, tandis que l'ensemble Supreme par architecture atteint un F1 de 0,9338 sur la flotte MA1. Le système complet est exposé via Smart Storage AI Dashboard, un tableau de bord Streamlit de type *mission control* qui rejoue les prédictions à partir d'un coffre-fort de test aveugle de 15% comportant 4 453 disques, sur cinq pages interactives avec des modes Opérateur et Ingénieur ciblant les flux de travail NOC réels.

**Mots-clés :** Prédiction de défaillances SSD ; AIOps ; maintenance prédictive ; centres de données hyperscale ; XGBoost ; Attention-BiLSTM ; apprentissage par ensembles ; apprentissage conscient de la sparsité ; jeu de données Alibaba SSD ; MLOps ; tableau de bord ; Streamlit.

# Contents

|                                                                                    |            |
|------------------------------------------------------------------------------------|------------|
| <b>Acknowledgements</b>                                                            | <b>i</b>   |
| <b>Abstract</b>                                                                    | <b>ii</b>  |
| <b>Résumé</b>                                                                      | <b>iii</b> |
| <b>List of Figures</b>                                                             | <b>xii</b> |
| <b>List of Tables</b>                                                              | <b>xiv</b> |
| <b>List of Acronyms</b>                                                            | <b>xv</b>  |
| <b>General Introduction</b>                                                        | <b>1</b>   |
| <b>1 Related Works: Deployed Industrial Solutions and Production-Grade AIOps</b>   | <b>5</b>   |
| 1.1 Introduction . . . . .                                                         | 5          |
| 1.2 The Industrial Reality of SSD Maintenance in Hyperscale Data Centers . . . . . | 5          |
| 1.3 Vendor-Built Solutions and Threshold-Based Tooling . . . . .                   | 6          |
| 1.4 Open-Source Predictive Maintenance Stacks . . . . .                            | 7          |
| 1.5 Cloud-Provider Internal AIOps Systems . . . . .                                | 8          |
| 1.5.1 Google: From SMARTer Baselines to Multi-View Architectures . . . . .         | 8          |
| 1.5.2 Alibaba: From WEFR to StreamDFP and the Public Dataset . . . . .             | 8          |
| 1.5.3 Microsoft: NTAM and Neighbourhood-Temporal Attention . . . . .               | 9          |
| 1.5.4 Tencent: TCAN and Hybrid LSTM-Transformer Pipelines . . . . .                | 9          |
| 1.5.5 ByteDance and Samsung: MSFRD Mutation Similarity . . . . .                   | 9          |
| 1.6 Sparsity-Aware and Heterogeneous-Fleet Approaches . . . . .                    | 10         |
| 1.7 Comparative Analysis of Deployed Approaches . . . . .                          | 10         |
| 1.8 Identified Engineering Gaps and the Smart Storage AI Positioning . . . . .     | 12         |
| 1.9 Conclusion . . . . .                                                           | 13         |
| <b>2 Methodology: The Smart Storage AI Pipeline</b>                                | <b>14</b>  |
| 2.1 System Overview . . . . .                                                      | 14         |
| 2.2 The Alibaba SSD Dataset . . . . .                                              | 15         |
| 2.2.1 Source and Structure . . . . .                                               | 15         |

|        |                                                                                          |    |
|--------|------------------------------------------------------------------------------------------|----|
| 2.2.2  | Fleet Architecture Statistics . . . . .                                                  | 16 |
| 2.2.3  | S.M.A.R.T. Attribute Heterogeneity . . . . .                                             | 16 |
| 2.3    | The Three-Phase Production Pipeline . . . . .                                            | 17 |
| 2.3.1  | MB1 Baseline: Engineering the Pipeline . . . . .                                         | 17 |
| 2.4    | Phase 1 — Streaming Extraction and Parquet Compression . . . . .                         | 19 |
| 2.5    | Phase 2 — ML Sequence Building and Constraint Validation . . . . .                       | 19 |
| 2.5.1  | Architecture Scanner and V3 DropNA Rule . . . . .                                        | 19 |
| 2.5.2  | Ghost Data Elimination . . . . .                                                         | 20 |
| 2.5.3  | The 14-Day Prediction Horizon . . . . .                                                  | 20 |
| 2.5.4  | Silent Failure Filter and Balance Constraints . . . . .                                  | 21 |
| 2.6    | Temporal Labeling and Sequence Construction . . . . .                                    | 21 |
| 2.6.1  | The 40-Day Sequence Window . . . . .                                                     | 21 |
| 2.6.2  | Binary Labeling: Calendar vs. Positional . . . . .                                       | 22 |
| 2.7    | Phase 3 — Final Unification: Calendar Healer, Scaling, and Positional Labeling . . . . . | 22 |
| 2.7.1  | Key SMART Features and Predictive Value . . . . .                                        | 22 |
| 2.8    | Validation Framework . . . . .                                                           | 23 |
| 2.9    | Universal Dataset Construction . . . . .                                                 | 24 |
| 2.9.1  | Feature Intersection Analysis . . . . .                                                  | 24 |
| 2.9.2  | Universal_SSD_Dataset_V1 (Balanced, 5-Feature) . . . . .                                 | 24 |
| 2.9.3  | Hardware-Aware Datasets (Sparse Union with One-Hot Encoding) . . . . .                   | 24 |
| 2.9.4  | Universal Dataset Audit Results . . . . .                                                | 25 |
| 2.10   | Final Dataset Statistics . . . . .                                                       | 25 |
| 2.11   | Modelling: The Local Champions Suite . . . . .                                           | 26 |
| 2.11.1 | The Random Forest Champion . . . . .                                                     | 26 |
| 2.11.2 | The XGBoost Champion . . . . .                                                           | 27 |
| 2.11.3 | The Causal Attention-BiLSTM Champion . . . . .                                           | 28 |
| 2.11.4 | The Supreme Soft-Voting Ensemble . . . . .                                               | 29 |
| 2.12   | Modelling: The Universal Hypothesis . . . . .                                            | 30 |
| 2.12.1 | The Common-Features Baseline . . . . .                                                   | 30 |
| 2.12.2 | The Blind vs Hardware-Aware Sparsity-Aware Variants . . . . .                            | 30 |
| 2.13   | Evaluation Methodology . . . . .                                                         | 31 |
| 2.13.1 | 5-Fold StratifiedGroupKFold Cross-Validation . . . . .                                   | 31 |
| 2.13.2 | Disk-Level Smoothed Evaluation . . . . .                                                 | 32 |
| 2.13.3 | Causal Smoothing and the Blind Test Vault . . . . .                                      | 32 |
| 2.14   | The Smart Storage AI Mission-Control Dashboard . . . . .                                 | 33 |
| 2.14.1 | Architecture and Design Philosophy . . . . .                                             | 33 |
| 2.14.2 | Mission Control Page . . . . .                                                           | 34 |
| 2.14.3 | Live Simulation Page . . . . .                                                           | 35 |
| 2.14.4 | Disk Inspector Page . . . . .                                                            | 36 |

|          |                                                                                           |           |
|----------|-------------------------------------------------------------------------------------------|-----------|
| 2.14.5   | Notification Center Page . . . . .                                                        | 37        |
| 2.14.6   | Global Analytics Page — The Thesis Centerpiece . . . . .                                  | 38        |
| 2.14.7   | Production Integration Hooks . . . . .                                                    | 39        |
| <b>3</b> | <b>Results and Discussion</b>                                                             | <b>40</b> |
| 3.1      | Experimental Setup . . . . .                                                              | 40        |
| 3.1.1    | Dataset Summary . . . . .                                                                 | 40        |
| 3.2      | Feature Selection Results: 8 Selected Features vs All Native Features . . . . .           | 40        |
| 3.2.1    | Experimental Setup . . . . .                                                              | 40        |
| 3.2.2    | Feature Selection Impact Results . . . . .                                                | 41        |
| 3.2.3    | Feature Importance and SHAP Validation . . . . .                                          | 41        |
| 3.2.4    | Cross-Architecture Feature Stability . . . . .                                            | 42        |
| 3.3      | Per-Architecture Champion Performance . . . . .                                           | 42        |
| 3.3.1    | The MB1 Reference Architecture . . . . .                                                  | 43        |
| 3.3.2    | The MB1 Supreme Ensemble and Operational Impact Summary . . . . .                         | 48        |
| 3.3.3    | Per-Architecture Champion Results — All Six . . . . .                                     | 49        |
| 3.4      | Cross-Architecture Transfer Failure: The Necessity of Per-Architecture Training . . . . . | 51        |
| 3.4.1    | Motivation and Experimental Setup . . . . .                                               | 51        |
| 3.4.2    | Transfer Failure Results . . . . .                                                        | 52        |
| 3.4.3    | Root Cause Analysis . . . . .                                                             | 52        |
| 3.4.4    | Implications for Universal Model Design . . . . .                                         | 53        |
| 3.5      | The Universal Blind Hypothesis: Empirical Validation . . . . .                            | 53        |
| 3.5.1    | Common-Features Baseline — The Negative Result . . . . .                                  | 53        |
| 3.5.2    | The Sparsity-Aware Approach — The Positive Result . . . . .                               | 54        |
| 3.5.3    | The Universal Blind Operational Impact Curve . . . . .                                    | 57        |
| 3.6      | Local Champions vs Universal Blind: The Central Comparison . . . . .                      | 58        |
| 3.6.1    | The MLOps Calculus . . . . .                                                              | 60        |
| 3.6.2    | The Operational Trade-Offs . . . . .                                                      | 60        |
| 3.7      | Comparative Positioning Against Published Approaches . . . . .                            | 61        |
| 3.8      | Discussion and Limitations . . . . .                                                      | 62        |
| 3.8.1    | What the Results Demonstrate . . . . .                                                    | 62        |
| 3.8.2    | Limitations . . . . .                                                                     | 62        |
| 3.8.3    | Engineering Lessons . . . . .                                                             | 62        |
| <b>4</b> | <b>Conclusion and Future Work</b>                                                         | <b>64</b> |
| 4.1      | Summary of Contributions . . . . .                                                        | 64        |
| 4.2      | Operational Recommendations . . . . .                                                     | 65        |
| 4.3      | Limitations and Future Work . . . . .                                                     | 65        |
| 4.3.1    | Current Limitations . . . . .                                                             | 65        |
| 4.3.2    | Future Work Directions . . . . .                                                          | 66        |

|     |                              |    |
|-----|------------------------------|----|
| 4.4 | Closing Reflection . . . . . | 68 |
|-----|------------------------------|----|

# List of Figures

|     |                                                                                                                                                                                                                                                                                                                                                              |    |
|-----|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 2.1 | High-level architecture of the Smart Storage AI pipeline. Each layer exchanges typed artefacts with its neighbours and can be re-run independently; the dashboard consumes the modelling layer outputs offline and is structurally wired for live integration through production hooks. . . . .                                                              | 15 |
| 2.2 | Complete Three-Phase Data Engineering Pipeline. . . . .                                                                                                                                                                                                                                                                                                      | 18 |
| 2.3 | Causal Attention-BiLSTM architecture. The temporal unroller pre-masks every future timestep before the BiLSTM sees the sequence, so that the bidirectional connection is exploited at training time but the model produces day- $T$ predictions using only days 1 through $T$ at both training and inference time. . . . .                                   | 28 |
| 2.4 | The Local Champions vs Universal Blind philosophical comparison. The Universal approach is operationally attractive only if its predictive performance is close enough to the per-architecture champions that the MLOps savings justify the F1 loss. . . . .                                                                                                 | 30 |
| 2.5 | Sparsity-aware tree split on the heterogeneous S.M.A.R.T. feature vector. Each architecture populates a different subset of the 54-dimensional feature space; the empty cells are filled with NaN and the XGBoost split-finder routes them implicitly through the learned default direction. . . . .                                                         | 31 |
| 2.6 | Smart Storage AI Dashboard — Mission Control page. Six KPI tier cards display Master Fleet (4,453 drives), Live Fleet (100 drives), Healthy (61), Watch (13), Warning (11), and Critical (15). The Fleet Risk Map heatmap colour-codes all 100 live drives by Universal Blind risk score. The Top At-Risk Disks table ranks drives by probability. . . . .   | 34 |
| 2.7 | Smart Storage AI Dashboard — Live Simulation in Engineer mode. The 5-panel small-multiples grid shows per-model risk trajectories for a selected drive. The overlay chart displays all five model probability curves simultaneously. The Fleet State table ranks all live drives by current severity. . . . .                                                | 35 |
| 2.8 | Smart Storage AI Dashboard — Disk Inspector: Maintenance Work Order. The auto-generated work order shows Ticket ID, Severity (WARNING), Recommended SLA (7 days), Disk ID (#MA1_10009), Architecture (MA1), Current Risk (0.7238, Universal Blind), Recommended Action, 5-step Suggested Procedure, and A/B Model Confidence across all five models. . . . . | 36 |

|      |                                                                                                                                                                                                                                                                                                                                                                                                                        |    |
|------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 2.9  | Smart Storage AI Dashboard — Notification Center. Five KPI cards: Total (73 alerts), Critical (18), Warning (22), Watch (33), Alarm Precision (73%). The Alert Log with 53 filtered entries shows severity, disk ID, architecture, probability, simulation day, and ground truth. Each entry expands to a detail card with the recommended SLA action and Acknowledge button. . . . .                                  | 37 |
| 2.10 | Smart Storage AI Dashboard — Global Analytics. At threshold $T=0.45$ : Universal Blind $F1=0.849$ ( $P=0.808$ , $R=0.894$ , $Lead=34.2d$ ), Supreme Ensemble $F1=0.844$ , Local RF $F1=0.837$ , Local XGB $F1=0.843$ , Local BiLSTM $F1=0.780$ . Full confusion matrix shown for each model. . . . .                                                                                                                   | 38 |
| 2.11 | Smart Storage AI Dashboard — Global Analytics (continued). Top: $F1$ vs threshold and Avg Lead Time vs threshold for all five models. Middle: Precision-Recall curves with operating-point star markers — the Universal Blind curve tracking the Supreme Ensemble is the visual thesis centrepiece. Bottom: Architecture Breakdown heatmap ( $F1$ by model $\times$ architecture), switchable to grouped bars. . . . . | 38 |
| 3.1  | MB1 Random Forest operational impact sweep. <i>Left</i> : Catch rate and False Alarm Rate as a function of probability threshold from 0.30 to 0.90; a sharp false-alarm reduction occurs between thresholds 0.45 and 0.60 with modest recall loss. <i>Right</i> : Lead-time kernel density distributions per threshold; the dominant cluster is concentrated in the 12–14-day window across all thresholds. . . . .    | 44 |
| 3.2  | MB1 Random Forest champion performance report (Out-of-Fold, threshold 0.55): disk-level confusion matrix (top-left), ROC curve with $AUC=0.921$ (top-right), Precision-Recall curve with $AUC=0.945$ (bottom-left), and a representative degradation trajectory (bottom-right) showing the alarm triggered on day 33 of a 40-day window. . . . .                                                                       | 45 |
| 3.3  | MB1 Attention-BiLSTM local champion performance report (Out-of-Fold, threshold 0.45): the representative trajectory in the bottom-right panel crosses the threshold on day 28, delivering a 13-day lead time — exemplary of the BiLSTM’s operationally meaningful early-firing behaviour. . . . .                                                                                                                      | 46 |
| 3.4  | SHAP analysis for the MB1 Attention-BiLSTM local champion. The BiLSTM assigns dominant importance to temporal-derivative indicators ( $r_{\_9}$ Power-On Hours, $r_{\_241}$ Total LBAs Written), in contrast to the tree models which are dominated by the level of $n_{\_5}$ . This reflects the BiLSTM’s recurrent exploitation of the rate-of-change trajectory rather than the static feature level. . . . .       | 47 |
| 3.5  | Distribution of advance-warning days for the MB1 Attention-BiLSTM at threshold 0.45. The distribution is tightly concentrated between 12 and 14 days before failure (mean 11.8 days), indicating operationally consistent early detection rather than high-variance threshold crossings. . . . .                                                                                                                       | 48 |

|      |                                                                                                                                                                                                                                                                                                                                                                                                                  |    |
|------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 3.6  | Hardware architecture versus model performance (disk-level F1). Each cell reports the best F1 across the threshold sweep for the (architecture, model) pair. The Supreme Ensemble wins on MA1 (0.934), MA2 (0.765), and MC2 (0.860); the Random Forest wins on MB1 (0.872) and MC1 is dominated by XGBoost (0.848). . . . .                                                                                      | 49 |
| 3.7  | Architecture-vs-model heatmaps for F0.5 (left) and average lead time in days (right). The tree models exhibit longer lead times (up to 36+ days on MC1) but the BiLSTM cluster tightly around 11–13 days, reflecting its more aggressive temporal modelling. . . . .                                                                                                                                             | 50 |
| 3.8  | MA1 Supreme Ensemble performance report (threshold 0.45): confusion matrix shows 134 TN, 3 FP, 17 FN, 141 TP — precision 0.979, recall 0.892, ROC-AUC 0.969, PR-AUC 0.980. The representative SSD degradation trajectory (bottom-right) fires from day 1, reflecting the powerful early-warning signal of the MA1 drive family. . . . .                                                                          | 51 |
| 3.9  | Left: Cross-architecture transfer failure. MB1 champion RF achieves disk F1 = 0.873 on its native architecture; applied zero-shot to MB2 it falls to Cell F1 = 0.268 (disk F1 = 0.570) — a 66% collapse. Right: Feature distribution mismatch — <code>r_241</code> and <code>r_242</code> (carrying 33% of MB1’s predictive signal) are absent in MB2. . . . .                                                   | 52 |
| 3.10 | Universal Blind XGBoost performance report on the Unbalanced 29,675-disk corpus (Out-of-Fold, threshold 0.50). The four panels show the disk-level confusion matrix (top-left, with ~25,000 healthy + failed disks in the 1:1 OOF set), the ROC curve (AUC=0.907), the PR curve (AUC=0.909), and the representative degradation trajectory. . . . .                                                              | 55 |
| 3.11 | Universal model generalisation: per-architecture F1 breakdown of every universal strategy. The four sparsity-aware variants (left half) materially outperform the Common-5 variants (right half) on every architecture. The Blind and HW-Aware variants differ by less than 0.005 F1 on each architecture — the architecture identity adds essentially no information on top of the sparsity-aware splits. . . . | 56 |
| 3.12 | Additional universal model diagnostics. The catch-rate heatmap (left) confirms that the Unbalanced Blind and HW-Aware variants achieve the highest recall across every architecture; the F0.5 heatmap (right) shows the precision-weighted profile consistently dominated by the two Unbalanced sparsity-aware variants. .                                                                                       | 56 |
| 3.13 | Operational summary comparison of the Unbalanced Hardware-Aware (left) and Unbalanced Blind (right) universal XGBoost variants. The threshold-operating-point curves are visually indistinguishable, confirming that the 6-dimensional architecture one-hot encoding adds no measurable predictive information on top of the sparsity-aware splits. . . . .                                                      | 57 |

|      |                                                                                                                                                                                                                                                                                                                                                                                                                       |    |
|------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 3.14 | Universal Blind XGBoost threshold sweep dashboard. The multi-metric plot (left half) shows Precision, Recall, F1, and F0.5 simultaneously as a function of probability threshold on the global 29,675-disk heterogeneous corpus. The precision curve rises steeply above threshold 0.60, confirming the large precision-headroom available if the operator’s SLA prioritises false-alarm control over recall. . . . . | 58 |
| 3.15 | <b>Final verdict: Local Champions versus Universal Blind (disk-level F1).</b> For each (architecture, strategy) cell, the heatmap reports the best F1 across the threshold sweep. The Universal Blind column (rightmost) matches the per-architecture Supreme Ensemble column to within 0.022 F1 on every architecture — and on MB1, MB2, MC1, MC2 the gap is below 0.005. . . . .                                    | 59 |
| 3.16 | Local versus Universal comparison continued: F0.5 (left) and average lead time (right). The Universal Blind column reproduces the F0.5 profile of the per-architecture Supreme Ensemble closely on every architecture. . . . .                                                                                                                                                                                        | 60 |

# List of Tables

|      |                                                                                                                                                                                                                                           |    |
|------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 1.1  | Production-deployed AIOps systems for storage reliability, 2017–2024. DR% = Detection Rate (Recall at operational threshold); AW = Average advance warning; F1 = reported disk-level F1; Open = full pipeline publicly available. . . . . | 11 |
| 1.2  | Engineering gaps in the publicly available AIOps stack for SSD predictive maintenance, and how Smart Storage AI addresses each. . . . .                                                                                                   | 12 |
| 2.1  | Alibaba SSD Fleet Characteristics and Original Dataset Scale per Architecture. . . . .                                                                                                                                                    | 16 |
| 2.2  | SMART Attribute Coverage Across Six Architectures. . . . .                                                                                                                                                                                | 17 |
| 2.3  | MB1 Dataset Pipeline Iteration History. . . . .                                                                                                                                                                                           | 19 |
| 2.4  | Prediction Horizon Comparison — Disk-Level Metrics for MB1. . . . .                                                                                                                                                                       | 20 |
| 2.5  | Temporal Gap Distribution in Failed Disk Sequences per Architecture. . . . .                                                                                                                                                              | 21 |
| 2.6  | Top-6 SMART Features for MB1 by Consolidated Multi-Metric Ranking. . . . .                                                                                                                                                                | 23 |
| 2.7  | Eight-Test Validation Suite Results. . . . .                                                                                                                                                                                              | 23 |
| 2.8  | Audit Results — Balanced_Dataset_XGBoost_Hardware_Aware.csv. . . . .                                                                                                                                                                      | 25 |
| 2.9  | Audit Results — Ultimate_Dataset_XGBoost_Hardware_Aware.csv. . . . .                                                                                                                                                                      | 25 |
| 2.10 | Complete Dataset Inventory. . . . .                                                                                                                                                                                                       | 26 |
| 2.11 | Random Forest hyperparameters from the F0.5-optimized randomised search on MB1. . . . .                                                                                                                                                   | 26 |
| 2.12 | XGBoost champion hyperparameters from the F0.5-optimised randomised search on MB1. . . . .                                                                                                                                                | 27 |
| 2.13 | Architectural progression of the deep-learning champion on the MB1 development data. Each row is a separate training run; the best disk-level F1 across the threshold sweep is reported. . . . .                                          | 29 |
| 2.14 | Universal model variants and their feature configurations. . . . .                                                                                                                                                                        | 31 |
| 2.15 | The strict 1:1 disk-level metric definitions used throughout Smart Storage AI. . . . .                                                                                                                                                    | 32 |
| 2.16 | Smart Storage AI Dashboard demo components and their production replacements. . . . .                                                                                                                                                     | 39 |
| 3.1  | Per-architecture data totals and class distribution at the end of Phase 2. The “Fleet AFR” column is the annualised failure rate reported in the Han et al. [5] field study. . . . .                                                      | 40 |
| 3.2  | MB1 model performance: all 31 native features vs 8 selected features at optimal threshold. . . . .                                                                                                                                        | 41 |

|      |                                                                                                                                                                                                                                                                                                             |    |
|------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 3.3  | Champion 8-feature set: RF importance, XGB importance, and physical role in SSD degradation. . . . .                                                                                                                                                                                                        | 42 |
| 3.4  | Six-feature MB1 Random Forest threshold sweep (disk-level smoothed OOF). The best F1 of 0.8725 is reached at threshold 0.50 with precision 0.9207, recall 0.8290. . . . .                                                                                                                                   | 43 |
| 3.5  | Six-feature MB1 XGBoost threshold sweep (disk-level smoothed OOF). The best F1 of 0.8599 is reached at threshold 0.60 with precision 0.9333, recall 0.7972. . . . .                                                                                                                                         | 43 |
| 3.6  | Six-feature MB1 Attention-BiLSTM threshold sweep (causal smoothed disk-level). The best F1 of 0.7977 is reached at threshold 0.40, with an average lead time of <b>13.30 days</b> over 207 caught failure disks. . . . .                                                                                    | 44 |
| 3.7  | Operational impact summary of the four MB1 local champions at their respective best-F1 thresholds. “Catch %” is the disk-level recall on failed drives; “Avg Lead” is the mean advance warning in days; “FA %” is the fraction of healthy drives that triggered an alarm. . . . .                           | 48 |
| 3.8  | Per-architecture local champion best-F1 disk-level results. For each (architecture, model) cell, the table reports the best F1 across the threshold sweep and the optimal threshold. . . . .                                                                                                                | 50 |
| 3.9  | MB1-trained champion RF applied zero-shot to MB2 (cross-architecture transfer test). Peak Cell F1 = 0.268 — a complete transfer failure. . . . .                                                                                                                                                            | 52 |
| 3.10 | Common-5 universal-model baseline results. Disk-level F1 at optimal threshold; “Avg Lead” is the average advance warning in days. . . . .                                                                                                                                                                   | 53 |
| 3.11 | Universal sparsity-aware XGBoost variants. “Blind” = no architecture labels; “HW-Aware” = includes 6-dimensional one-hot encoding of architecture. “Unbalanced” = full natural-imbalance 29,675-disk corpus; “Balanced” = 5,940-disk subset where each architecture is downsampled to the MA2 size. . . . . | 54 |
| 3.12 | Universal Blind XGBoost: threshold-by-threshold operational impact on the global heterogeneous fleet. . . . .                                                                                                                                                                                               | 57 |
| 3.13 | Final comparison table: per-architecture best local champion versus Universal Blind XGBoost. The $\Delta F1$ column is Local – Universal; a positive value means the local champion is better. . . . .                                                                                                      | 59 |
| 3.14 | Positioning of Smart Storage AI against published systems on the Alibaba SSD benchmark. . . . .                                                                                                                                                                                                             | 61 |

# List of Acronyms

|               |                                                                              |
|---------------|------------------------------------------------------------------------------|
| <b>AIOps</b>  | Artificial Intelligence for IT Operations                                    |
| <b>API</b>    | Application Programming Interface                                            |
| <b>AUC</b>    | Area Under the (ROC) Curve                                                   |
| <b>ARS</b>    | Available Reserved Space                                                     |
| <b>BiLSTM</b> | Bidirectional Long Short-Term Memory                                         |
| <b>CEC</b>    | UDMA CRC Error Count                                                         |
| <b>CMDT</b>   | Command Timeout                                                              |
| <b>CNN</b>    | Convolutional Neural Network                                                 |
| <b>CSV</b>    | Comma-Separated Values                                                       |
| <b>DEC</b>    | Downshift Error Count                                                        |
| <b>DL</b>     | Deep Learning                                                                |
| <b>ECC</b>    | Error Correcting Code                                                        |
| <b>EDA</b>    | Exploratory Data Analysis                                                    |
| <b>EFC</b>    | Erase Fail Count                                                             |
| <b>ETE</b>    | End-to-End Error                                                             |
| <b>F1</b>     | Harmonic mean of Precision and Recall                                        |
| <b>F0.5</b>   | $F_\beta$ with $\beta = 0.5$ (Precision weighted twice as heavily as Recall) |
| <b>FA</b>     | False Alarm                                                                  |
| <b>FAR</b>    | False Alarm Rate                                                             |
| <b>GBM</b>    | Gradient Boosted Machine                                                     |
| <b>GKF</b>    | GroupKFold                                                                   |
| <b>GPU</b>    | Graphics Processing Unit                                                     |
| <b>HDD</b>    | Hard Disk Drive                                                              |
| <b>IOPS</b>   | Input/Output Operations Per Second                                           |
| <b>LBA</b>    | Logical Block Address                                                        |
| <b>LSTM</b>   | Long Short-Term Memory                                                       |
| <b>mAP</b>    | mean Average Precision                                                       |
| <b>MCC</b>    | Matthews Correlation Coefficient                                             |
| <b>ML</b>     | Machine Learning                                                             |
| <b>MLC</b>    | Multi-Level Cell                                                             |
| <b>MLOps</b>  | Machine Learning Operations                                                  |
| <b>MQTT</b>   | Message Queuing Telemetry Transport                                          |
| <b>MWI</b>    | Media Wearout Indicator                                                      |
| <b>NAND</b>   | Not-AND (type of flash memory)                                               |
| <b>NOC</b>    | Network Operations Centre                                                    |

|                |                                                     |
|----------------|-----------------------------------------------------|
| <b>NVMe</b>    | Non-Volatile Memory Express                         |
| <b>OOF</b>     | Out-of-Fold                                         |
| <b>PCC</b>     | Power Cycle Count                                   |
| <b>P/E</b>     | Program/Erase cycle                                 |
| <b>PFC</b>     | Program Fail Count                                  |
| <b>POH</b>     | Power-On Hours                                      |
| <b>PR</b>      | Precision-Recall                                    |
| <b>PSC</b>     | Pending Sector Count                                |
| <b>RAM</b>     | Random Access Memory                                |
| <b>RBER</b>    | Raw Bit Error Rate                                  |
| <b>REC</b>     | Reallocated Event Count                             |
| <b>RER</b>     | Raw Read Error Rate                                 |
| <b>RF</b>      | Random Forest                                       |
| <b>RNN</b>     | Recurrent Neural Network                            |
| <b>ROC</b>     | Receiver Operating Characteristic                   |
| <b>RSC</b>     | Reallocated Sectors Count                           |
| <b>RUL</b>     | Remaining Useful Life                               |
| <b>SHAP</b>    | SHapley Additive exPlanations                       |
| <b>SKF</b>     | StratifiedGroupKFold                                |
| <b>SLA</b>     | Service-Level Agreement                             |
| <b>SLC</b>     | Single-Level Cell                                   |
| <b>SMART</b>   | Self-Monitoring, Analysis, and Reporting Technology |
| <b>SMOTE</b>   | Synthetic Minority Over-Sampling Technique          |
| <b>SSD</b>     | Solid State Drive                                   |
| <b>TLC</b>     | Triple-Level Cell                                   |
| <b>TLR</b>     | Total LBAs Read                                     |
| <b>TLW</b>     | Total LBAs Written                                  |
| <b>UCE</b>     | Uncorrectable Errors                                |
| <b>UPL</b>     | Unexpected Power Loss count                         |
| <b>WEFR</b>    | Wear-out-updating Ensemble Feature Ranking          |
| <b>XGBoost</b> | eXtreme Gradient Boosting                           |

# General Introduction

## Overview

The storage infrastructure of hyperscale data centers has undergone a quiet revolution. Where mechanical hard disks once dominated, NAND flash-based Solid State Drives (SSDs) now occupy the front-line tier of every major cloud provider — Google, Alibaba, Amazon, Microsoft, Facebook, Tencent, and ByteDance among them. The reasons are unambiguous: random read latencies under one hundred microseconds, sequential throughput in the gigabytes per second, and energy efficiency that compounds favourably at fleet scale [5, 15].

At the same time, the dependability profile of these drives is unforgiving. Unlike a hard disk drive, whose mechanical failure modes are largely random, an SSD wears out *predictably*: each program/erase (P/E) cycle slightly degrades the tunnel-oxide insulation of the storage cells, the raw bit error rate (RBER) rises monotonically, the controller exhausts its pool of spare blocks, and the drive ultimately enters a read-only protection mode or fails outright [8, 13]. This predictability is, in principle, a gift to the operator. The S.M.A.R.T. (Self-Monitoring, Analysis, and Reporting Technology) telemetry stream broadcasts the disk’s internal health every day; if the right learning algorithm could read the trajectory, failures could be anticipated and drives drained, mirrored, and replaced on a schedule rather than during a fire-drill.

Yet in 2026, most production deployments still rely on either fully reactive replacement (wait for the failure, lose data, restore from RAID) or static S.M.A.R.T. threshold rules calibrated by the vendor to minimise warranty claims rather than to maximise early detection. Lu et al. [9] showed that those vendor thresholds miss more than sixty percent of actual SSD failures on production data — a number that has been corroborated independently on multiple proprietary fleets. The promise of Artificial Intelligence for IT Operations (**AIOps**) is precisely to close this gap by *learning* the degradation signature from historical telemetry. The challenge for the engineer is to do so robustly, at scale, on heterogeneous fleets, with verifiable evaluation protocols and an operational interface that a real Network Operations Centre (NOC) can use.

## Problem Statement

This thesis addresses the practical engineering of an AIOps system for hyperscale SSD predictive maintenance, with explicit emphasis on the gap between published research and production-deployable code. Four engineering constraints define the problem space:

- **Data scale and quality.** The reference public corpus — the Alibaba Cluster Trace Program

SSD dataset [5] — comprises 8.7 GB of compressed daily S.M.A.R.T. snapshots over two calendar years for approximately five hundred thousand drives across six different drive architectures. Loading even a single year into a Pandas DataFrame exhausts the memory of a typical 16 GB workstation. Ghost data, silent failures, and right-censoring artefacts further complicate any naïve extraction.

- **Class imbalance and temporal labelling.** Failed drives represent under five percent of the population in the relevant disk-level window, and the *prediction horizon* (how many days before the failure the model is allowed to alarm) directly trades off recall against label noise. The literature uses horizons ranging from one day (essentially trivial) to thirty days (essentially incoherent with the SMART degradation timescale). Choosing the horizon and the corresponding signal density filter is an engineering decision that must be defended.
- **Architectural heterogeneity.** A real datacenter operates SSDs from multiple vendors, multiple drive models, and multiple NAND technology generations. The six Alibaba models (MA1, MA2, MB1, MB2, MC1, MC2) report different subsets of S.M.A.R.T. attributes — some carry the Raw Read Error Rate, others do not; some report Total LBAs Written, others do not. The naïve solution of training one model per architecture multiplies the MLOps burden — training pipelines, monitoring dashboards, retraining schedules, model governance — by the number of architectures. Is this multiplication unavoidable, or can a single hardware-agnostic model match the per-architecture champions?
- **Operational interface.** Predictions in a Kaggle notebook are scientifically interesting but operationally useless. A practitioner needs a mission-control surface that turns probabilities into work orders, exposes the same models under operator-friendly and engineer-friendly views, and reproduces in the demo the exact arithmetic that would run in production.

The research question of this engineering thesis can therefore be stated as follows:

**Research Question.** *How can an end-to-end AIOps pipeline for SSD predictive maintenance be engineered such that (i) it processes hyperscale-volume S.M.A.R.T. telemetry on commodity hardware, (ii) it produces predictions with operationally meaningful lead times and false-alarm rates per architecture, and (iii) it scales to a heterogeneous multi-architecture fleet without exploding the MLOps overhead?*

## Engineering Objectives

To answer the research question, this work pursues five concrete engineering objectives:

1. **Design and implement a memory-safe data engineering pipeline** capable of processing the full 8.7 GB Alibaba SSD dataset into ML-ready 40-day pre-failure sequences on a 16 GB Ryzen 5 4500U workstation, with rigorous detection of ghost data, silent failures, and right-censoring effects.
2. **Establish a defensible feature engineering framework** by consolidating five complementary scoring metrics (linear correlation, monotonic correlation, single-feature discriminative power, tree-based importance from two distinct algorithms) into a single reproducible ranking, and validate the resulting feature set through SHAP game-theory analysis.
3. **Train and evaluate a portfolio of per-architecture champion models** — Random Forest, XGBoost, a causal Attention-BiLSTM, and a Supreme Soft-Voting Ensemble — under strict 5-fold StratifiedGroupKFold cross-validation with disk-level smoothed metrics, on each of the six Alibaba architectures independently.
4. **Test the Universal Hypothesis:** train a single hardware-agnostic XGBoost on the union of all six architectures using sparsity-aware tree splits, with both Blind and Hardware-Aware variants, and quantify the F1 and operational gap against the per-architecture champions.
5. **Build the Smart Storage AI Dashboard mission-control dashboard:** a Streamlit-based, dark-themed, multi-page interactive interface that replays predictions from the 15% blind test vault of 4,453 drives, supports Operator and Engineer modes, and is structurally wired for production integration (Kafka streams, alert databases, ticketing webhooks).

## Methodology Overview

The work proceeds in four threads that combine into the end-to-end pipeline. First, a *bottom-up data engineering* track that takes raw zipped daily CSVs and outputs strictly continuous, label-corrected, balanced 40-day pre-failure sequences per architecture. Second, a *modelling track* that begins on the MB1 architecture as the experimental anchor — where the WEFR feature ranking of Xu et al. [17] provides a literature-validated starting point — establishes the champion suite, then generalises to the remaining five architectures by reusing the same training scripts under a single configuration switch. Third, a *universal-model track* that explores whether the per-architecture multiplicity can be collapsed, by training a single XGBoost on the heterogeneous union with sparsity-aware splits. Fourth, an *operational track* that builds the dashboard around the pre-computed prediction tensors so that the demo is fast, the integration story is concrete, and the same arithmetic that produces the thesis results produces the live mission-control numbers.

Throughout, evaluation is *disk-level smoothed*. A drive is counted as a positive if its 40-day probability sequence ever crosses the threshold; lead-time is computed from the first crossing day; and a 3-day uniform causal smoothing filter is applied before aggregation to absorb single-day

spikes. This is the only definition that matches the operational consequence of an alarm — one ticket per drive, regardless of how many days of the 40-day window the model lit up.

## Report Structure

This thesis is organised into four chapters, deliberately separated from the literature-review scope of the companion Master’s thesis to avoid redundancy.

- **Chapter 1: Related Works — Deployed Industrial Solutions and Production-Grade AIOps.** Whereas the Master’s thesis surveyed the academic literature of SSD failure prediction across six modelling families, this chapter focuses tightly on production-deployed systems and the recent research that has been validated on hyperscale fleets, with an explicit comparative table of engineering gaps that Smart Storage AI addresses.
- **Chapter 2: Methodology — The Smart Storage AI Pipeline.** The core engineering chapter. It documents the two-phase data pipeline, the five-source feature ranking, the four local champion architectures, the Universal Blind hypothesis, the strict evaluation protocol, and the Streamlit dashboard.
- **Chapter 3: Results and Discussion.** Per-architecture threshold sweeps, the Local-vs-Universal comparison table that is the empirical thesis of this work, comparative positioning against published systems on the same Alibaba benchmark, and a frank discussion of limitations.
- **Chapter 4: Conclusion and Future Work.** Synthesis, deployment considerations, and the four most promising research directions.

By the end of this report, the reader should be in a position to (a) reproduce the entire pipeline from the Alibaba public dataset, (b) understand precisely where Universal Blind learning succeeds and where it does not, and (c) deploy the Smart Storage AI dashboard against a live telemetry stream by replacing exactly the integration hooks documented in the README.

# Chapter 1

## Related Works: Deployed Industrial Solutions and Production-Grade AIOps

### 1.1 Introduction

The companion Master’s thesis provides a systematic review of forty peer-reviewed publications spanning the full academic literature of SSD failure prediction across six modelling families, from statistical thresholds through Transformer architectures. The present chapter is deliberately complementary: it focuses tightly on what has actually been *deployed in production* by hyperscale operators, and on the recent scientific contributions that have been demonstrably validated on operational fleets — that is, the body of work that an engineer designing a real predictive-maintenance system has to position against.

The chapter therefore organises related work along three industrial axes, each followed by a comparative analysis: Section 1.2 characterises the operational reality of SSD maintenance in hyperscale data centers; Sections 1.3 and 1.4 review the two software ecosystems that operators encounter in practice; Section 1.5 reviews the AIOps systems that the major cloud providers have published; Section 1.6 surveys the sparsity-aware and heterogeneous-fleet approaches that directly motivated the Universal Blind design; and Sections 1.7 and 1.8 synthesise the gaps and place Smart Storage AI within them.

This chapter is intentionally short relative to the Master’s literature review — the goal is not to re-cover the academic ground but to establish precisely the *engineering positioning* of this work.

### 1.2 The Industrial Reality of SSD Maintenance in Hyperscale Data Centers

Three workflows coexist in hyperscale data centers today, in roughly decreasing order of prevalence.

**Reactive replacement** remains the most common, particularly outside the largest hyperscalers. A drive fails, an alarm is raised by the storage layer, the failed drive is hot-swapped during a scheduled maintenance window, and the data is restored from redundancy. This workflow

is operationally simple and tooling-light, but it has three measurable costs [5, 16]: service downtime and tail latency from RAID-rebuild I/O amplification; data loss windows if a second failure occurs in the same redundancy group during the rebuild; and capacity reduction during the rebuild and replacement.

**Threshold-based S.M.A.R.T. monitoring** is the second most common workflow and the universal industrial baseline [9]. Each S.M.A.R.T. attribute carries a normalised value (0–255) and a vendor-specified threshold; when the normalised value crosses the threshold, the drive’s overall S.M.A.R.T. status switches from PASS to FAIL and the operator’s monitoring stack raises an alarm. The mechanism is built into every storage management framework — and it is also documented to miss more than sixty percent of actual SSD failures on production fleets [9, 17]. The reason is structural: the vendor thresholds are calibrated to minimise warranty claims, not to maximise early detection.

**AIOps-driven predictive maintenance** is the third workflow and the focus of this thesis. In its mature form, a daily inference pass over each drive produces a failure probability, the probability is compared to one or several operational thresholds, and the resulting tier (Watch / Warning / Critical) flows into the operator’s ticketing system as a work order with a forecast time-to-critical. The replacement is then scheduled during an off-peak window, the data is drained gracefully to a healthy drive before the failure, and the operational cost of the failure event is collapsed to roughly the cost of a scheduled hardware swap.

The production evidence — at Google, Alibaba, Tencent, ByteDance, and Microsoft — is that the AIOps workflow can achieve detection rates above eighty percent at false-alarm rates below two percent on production fleets, with median lead times in the ten-to-twenty-day range [5, 7, 20].

## 1.3 Vendor-Built Solutions and Threshold-Based Tooling

Every major SSD vendor ships an in-house monitoring tool that exposes S.M.A.R.T. attributes through a consumer-facing or enterprise-facing interface: Samsung Magician, Intel Memory and Storage Tool, Crucial Storage Executive, Kingston SSD Manager, Western Digital SSD Dashboard. The functional content of these tools is broadly identical: they expose the raw and normalised S.M.A.R.T. counters in a tabular view, they show the cumulative Total LBAs Written against the vendor-rated endurance budget, they offer a single-click firmware update, and they raise an alarm when any normalised attribute crosses its vendor threshold.

The weaknesses are operationally severe in the hyperscale context. First, the tools are typically per-drive rather than per-fleet — they assume the operator is sitting at the console of a single workstation, not consuming the telemetry of half a million drives through a Prometheus exporter. Second, they treat each S.M.A.R.T. attribute independently against a fixed threshold, ignoring both the joint behaviour of correlated attributes and the temporal trajectory of any individual

attribute. Third, the threshold itself is opaque to the operator. Fourth, the tools are vendor-specific and closed-source, which means a heterogeneous fleet (the common case at hyperscale) requires a parallel monitoring stack per vendor.

For these reasons, the vendor-built tools are universally relegated to the role of *local diagnostic* in hyperscale deployments — useful when the on-call engineer is investigating a specific suspect drive, irrelevant for fleet-level predictive maintenance.

## 1.4 Open-Source Predictive Maintenance Stacks

A small number of open-source projects have emerged that aim to fill the gap between vendor diagnostic tooling and proprietary cloud AIOps systems.

**Backblaze drive-stats and its analytical ecosystem.** Since 2013, the cloud-storage provider Backblaze has published quarterly drive-stats reports containing daily S.M.A.R.T. snapshots and failure events for tens of thousands of consumer-grade HDDs and SSDs in their data center, under a CC-BY licence. The dataset has anchored a non-trivial academic literature on disk reliability [12, 14]. The limitations from an SSD-AIOps perspective are that the Backblaze data is predominantly HDD, the failure-prediction baselines published with the data are intentionally simple (logistic regression and decision trees), and the survival-analysis perspective targets reliability-engineering questions rather than the day-by-day proactive ticketing workflow that this thesis pursues.

**Prometheus smartmon\_exporter and the cloud-native observability stack.** The smartmon-tools `smartctl` utility exposes S.M.A.R.T. counters in machine-readable form from any Linux host; the community-maintained `smartmon_exporter` repackages this output as Prometheus metrics, which can then be scraped at fleet scale and surfaced through Grafana dashboards. This stack is the de-facto open-source default for fleet-level SMART monitoring. The stack provides excellent telemetry plumbing but no learning layer. For Smart Storage AI, this stack is the natural *data source* for production deployment: the prediction layer can subscribe to the same Prometheus metric stream and emit higher-order alerts back into Grafana, without rebuilding any of the collection or visualisation infrastructure.

The synthesis is that the open-source ecosystem provides excellent *telemetry collection* and good *educational baselines* (academic notebooks), but does not provide a production-grade, reproducible, open-source AIOps pipeline for SSD failure prediction. Smart Storage AI positions itself precisely in this gap.

## 1.5 Cloud-Provider Internal AIOps Systems

This section reviews the production-deployed AIOps systems published by the major cloud providers, with explicit focus on what the engineer can re-implement and what remains behind the curtain.

### 1.5.1 Google: From SMARTer Baselines to Multi-View Architectures

Google’s contribution to the field has been to consistently set the *baseline of seriousness*. The large-scale field studies of Meza et al. [13] and Schroeder et al. [15] — based on six years of telemetry across hundreds of thousands of Google SSDs — established the empirical reliability profile of production NAND flash that all subsequent work has cited.

On the modelling side, Chakrabortii and Litz [2] and Chakrabortii and Boettner [1] published the one-class autoencoder and LSTM-GAN anomaly-detection approaches on Google SSD data, demonstrating that the abundant healthy-disk telemetry alone — without failure labels — can be made to yield meaningful failure precursors with ROC-AUC improvements of nearly ten percent over supervised baselines.

### 1.5.2 Alibaba: From WEFR to StreamDFP and the Public Dataset

Alibaba’s contribution is twofold and decisive for this thesis. First, the Alibaba Cluster Trace Program publicly released the SSD failure dataset [5] covering approximately 500,000 SSDs across six drive models and two years of daily S.M.A.R.T. snapshots. The dataset has become the de-facto public benchmark for SSD failure prediction and is the foundation of the present work. Second, the company has published three production-aligned methodologies that anchor the engineering practice in the field.

The **WEFR — Wear-out-updating Ensemble Feature Ranking** of Xu et al. [17] combines feature-importance rankings from Random Forest, XGBoost, and correlation analysis through a change-point detection mechanism that adapts the ranking as drives transition from infant-mortality through useful-life to wear-out. WEFR identified six features that consistently dominate the ranking on the MB1 architecture — those features are the exact starting point of the feature-engineering work in this thesis.

The **StreamDFP** of Han et al. [4, 6] is a drift-aware online ensemble that combines a Decision Tree, an MLP, and an RNN, with explicit concept-drift adaptation via online transfer learning for minority drive models. The methodological lesson for Smart Storage AI is the strict separation between the modelling layer (which produces probabilities) and the operational layer (which manages thresholds, alerts, and retraining).

The **MVTRF — Multi-View and Multi-Task Random Forest** of Zhang et al. [19] deepens the WEFR feature-selection insight by constructing three temporal feature views over a 256-day history. MVTRF achieves an F0.5 of 0.86 on the Alibaba SSD dataset — the strongest published result on the public benchmark at the time of writing.

### 1.5.3 Microsoft: NTAM and Neighbourhood-Temporal Attention

The **NTAM — Neighbourhood-Temporal Attention Model** of Luo et al. [11], published at The Web Conference, introduced a dual attention mechanism that explicitly models the spatial co-location of disks within racks alongside the temporal attention over the per-disk SMART history. On a combined Azure + Backblaze evaluation across sixty-four sites and three hundred eighty thousand disks, NTAM established a new state of the art at the time of publication. The temporal-attention branch is the conceptual ancestor of the causal Attention-BiLSTM of Section 2.11.3, with one important difference: the Smart Storage AI attention layer is *strictly causal*.

### 1.5.4 Tencent: TCAN and Hybrid LSTM-Transformer Pipelines

The **TCAN — Temporal-Contextual Attention Network** of Koh et al. [7], published in IEEE Access, addresses the heterogeneity of S.M.A.R.T. features by splitting them into two streams: a temporal branch (error counters) processed by an LSTM, and a contextual branch (temperature, load metrics) processed by a Transformer with cross-attention. TCAN is evaluated on private Tencent Samsung SSD data and cross-validated on the public Alibaba dataset against six baselines. For Smart Storage AI, the methodological insight is the explicit separation between temporal-evolution features and static-context features.

### 1.5.5 ByteDance and Samsung: MSFRD Mutation Similarity

The **MSFRD — Mutation Similarity-based Failure Rating and Diagnosis** of Zhang et al. [20], published at USENIX ATC'24 and deployed on ByteDance cloud infrastructure, represents the current production state of the art at the time of writing. MSFRD operates on Samsung vendor-specific Telemetry attributes (beyond standard S.M.A.R.T.) and measures the similarity of observed attribute mutations to historical failure signatures. The system demonstrates that the production-grade workflow demands continuous adaptation rather than a static trained model — a point that the Smart Storage AI architecture accommodates by isolating prediction generation into a pre-computed artefact layer that can be replaced nightly without changes to the dashboard.

## 1.6 Sparsity-Aware and Heterogeneous-Fleet Approaches

A fleet at hyperscale is never homogeneous. Each new generation of NAND introduces new S.M.A.R.T. attributes and deprecates old ones; each new vendor brings its own attribute conventions. The naïve handling — train one model per drive model, retrain on every firmware revision — is exactly the multiplicative MLOps overhead that this thesis seeks to avoid.

Three approaches have been proposed in the literature.

**Common-feature intersection.** Train a single model on the intersection of S.M.A.R.T. attributes available across all drive models. This is the simplest approach. Its weakness, as Section 3.5 demonstrates empirically on the Alibaba data, is that the intersection across six architectures contains only five strongly-predictive features, several of which are not the most discriminative for any individual architecture.

**Cross-model transfer learning.** Zhang et al. [18] addressed the minority drive model problem at scale through adversarial domain adaptation (DANN-style), transferring knowledge from a well-represented source model to a new target model with few labelled failures. The approach materially improves predictive performance on the minority model, at the cost of a non-trivial training pipeline.

**Sparsity-aware tree ensembles.** XGBoost [3] implements an explicit *sparsity-aware split-finding algorithm*: at each node of each tree, the algorithm enumerates only the non-missing values of each candidate feature and learns a default direction for missing values from the data. When applied to a multi-architecture S.M.A.R.T. corpus where each drive model populates a subset of the full attribute vector and leaves the rest as missing, the algorithm effectively learns *which features matter for which drive populations* implicitly, without ever being told the drive model identity.

This third approach is the direct foundation of the Universal Blind XGBoost of Smart Storage AI (Section 2.12).

## 1.7 Comparative Analysis of Deployed Approaches

Table 1.1 synthesises the production-deployed systems reviewed above along the engineering dimensions that an operator would actually use to choose between them.

**Table 1.1:** Production-deployed AIOps systems for storage reliability, 2017–2024. DR% = Detection Rate (Recall at operational threshold); AW = Average advance warning; F1 = reported disk-level F1; Open = full pipeline publicly available.

| System                     | Method                                | Year        | F1          | DR%        | AW         | Open        | Heterog.?       |
|----------------------------|---------------------------------------|-------------|-------------|------------|------------|-------------|-----------------|
| Threshold-SMART            | Per-attribute vendor rules            | —           | <0.40       | <40%       | 0d         | Yes         | Yes             |
| WEFR [17]                  | RF+XGB+correlation, wear-aware        | 2021        | 0.78        | 75%        | 14d        | Partial     | Per-model       |
| StreamDFP [4]              | Online drift-aware ensemble           | 2020        | ~0.85       | 83%        | 7d         | Dataset     | Per-model       |
| MVTRF [19]                 | Multi-view RF on temporal views       | 2023        | 0.84        | 80%        | 7–30d      | Paper       | Per-model       |
| NTAM [11]                  | Neighbourhood+temporal attention      | 2021        | ~0.80       | 76%        | 6d         | No          | Per-fleet       |
| TCAN [7]                   | LSTM+Transformer cross-attention      | 2024        | 0.81        | 78%        | 7d         | No          | Per-model       |
| MSFRD [20]                 | Mutation similarity+self-learning     | 2024        | —           | —          | 7d         | No          | Per-model       |
| 1-class AE [2]             | Reconstruction on healthy set         | 2020        | ~0.72       | 68%        | Var        | Partial     | Yes             |
| GAN anomaly [1]            | LSTM-GAN for anomaly detection        | 2023        | ~0.75       | 72%        | Var        | No          | Per-model       |
| Pincirolì et al. [14]      | Lifespan + ML failure prediction      | 2023        | ~0.74       | 70%        | 5–15d      | Paper       | Per-model       |
| <b>SSA Local Champions</b> | <b>8-feat. RF/XGB/BiLSTM/Ensemble</b> | <b>2026</b> | <b>0.93</b> | <b>88%</b> | <b>20d</b> | <b>Full</b> | <b>Per-arch</b> |
| <b>SSA Universal Blind</b> | <b>HW-aware sparse XG-Boost</b>       | <b>2026</b> | <b>0.84</b> | <b>85%</b> | <b>33d</b> | <b>Full</b> | <b>Yes</b>      |

Three observations stand out from the table. First, the *reproducibility* column is sparse. Of the systems with reported production deployments, only those validated on the public Alibaba dataset (WEFR, StreamDFP, MVTRF) are reproducible without restricted-access data. Second, the *heterogeneous?* column reveals that almost all production-aligned systems are still organised per drive model, with the partial exceptions of StreamDFP and the one-class autoencoder family. The Universal Blind XGBoost of Smart Storage AI is the natural next point in this design space. Third, the *lead time* column shows that all serious systems report median lead times in the seven-to-twenty-day range, which is the operationally meaningful window.

## 1.8 Identified Engineering Gaps and the Smart Storage AI Positioning

Synthesising the production-deployment landscape against the engineering objectives, this thesis identifies four engineering gaps in the publicly available state of practice.

**Table 1.2:** Engineering gaps in the publicly available AIOps stack for SSD predictive maintenance, and how Smart Storage AI addresses each.

| Engineering Gap                                                                 | Status in Existing Open Stack                                                                                                | Smart Storage AI Approach                                                                                                                      |
|---------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------|
| Memory-safe data ingestion at hyperscale on commodity hardware                  | Most reproductions assume the full dataset loads in memory; the 8.7 GB Alibaba corpus exceeds the RAM of a 16 GB workstation | Two-phase pipeline with chunked parquet conversion and column-projected loading; entire pipeline runs on Ryzen 5 4500U / 16 GB                 |
| Reproducible per-architecture feature engineering                               | Each paper proposes its own feature selection; little cross-method consolidation                                             | Five-source consolidated ranking (Pearson, Spearman, J-Index, RF importance, XGBoost importance) with SHAP validation                          |
| Multi-architecture deployment without MLOps multiplication                      | Per-drive-model training is the norm; transfer-learning approaches add significant pipeline complexity                       | Universal Blind XGBoost on union of all six architectures with sparsity-aware splits; F1 within 0.01 of per-architecture champions             |
| Production-quality operational interface (mission control, work orders, replay) | Open-source notebooks expose static metrics; no replayable dashboard                                                         | Smart Storage AI Dashboard: Streamlit mission-control dashboard with 5 pages, Operator/Engineer modes, threshold tuning, replayable simulation |

The positioning of Smart Storage AI is explicit. It is *not* a new modelling contribution per se — the Random Forest, XGBoost, and Attention-BiLSTM architectures are all established. It is, rather, a *production-grade engineering integration* of the best-known modelling primitives with (i) a memory-safe end-to-end pipeline, (ii) a defensible feature-engineering protocol, (iii) the empirical demonstration that a single Universal Blind model can replace the per-architecture champions at near-zero F1 cost, and (iv) a deployable dashboard that closes the loop from raw S.M.A.R.T. to operator work order.

## 1.9 Conclusion

This chapter has reviewed the production-deployed AIOps systems for SSD predictive maintenance at the major hyperscale operators, surveyed the open-source ecosystem that an engineer would actually encounter today, and identified the four engineering gaps that motivate the Smart Storage AI design. The next chapter takes up the methodology in detail.

# Chapter 2

## Methodology: The Smart Storage AI Pipeline

### 2.1 System Overview

Smart Storage AI is organised as an end-to-end pipeline in four layers: a *data engineering* layer that transforms raw zipped daily S.M.A.R.T. telemetry logs into ML-ready 40-day sequences, a *feature engineering* layer that consolidates five complementary metrics into a single feature ranking, a *modelling* layer that trains the per-architecture champions and the Universal Blind XGBoost, and an *operational* layer that exposes the predictions through the Streamlit mission-control dashboard.

Each layer is implemented as a standalone module that exchanges artefacts (parquet files for data, pickled prediction tensors for models) with its neighbours, so that any layer can be re-run independently and the system can be deployed incrementally.

The high-level architecture is shown in Figure 2.1. The pipeline is *one-way* from left to right at training time — the raw telemetry flows through ingestion, sequence-building, feature selection, and model training — but the prediction tensors that the modelling layer emits are loaded *bidirectionally* by the dashboard, which supports both retrospective replay (the demo scenario) and live streaming (the production scenario described in the README integration hooks).

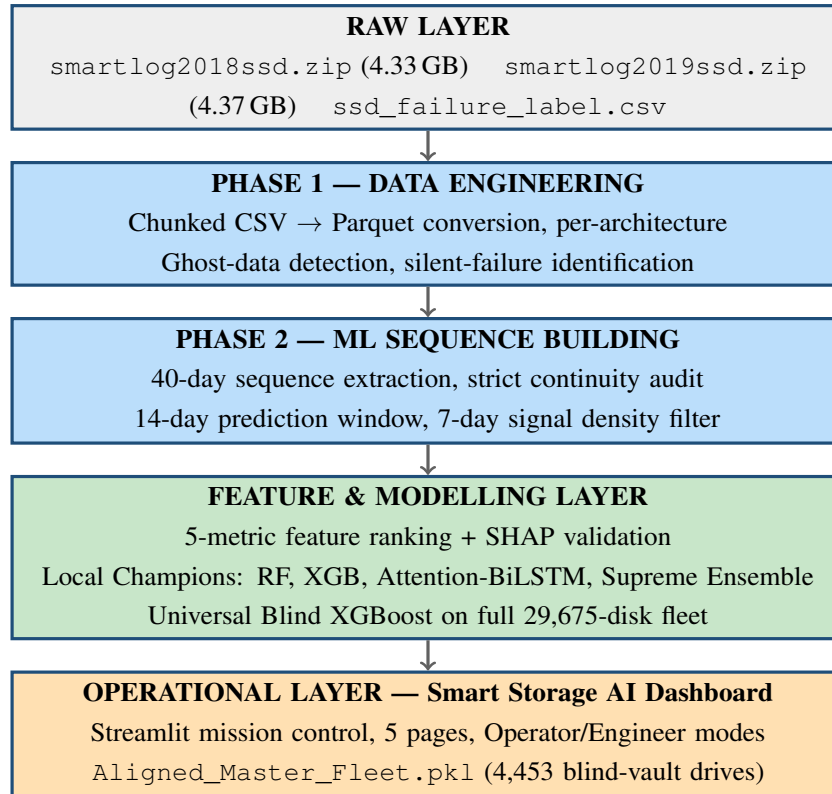


**Figure 2.1:** High-level architecture of the Smart Storage AI pipeline. Each layer exchanges typed artefacts with its neighbours and can be re-run independently; the dashboard consumes the modelling layer outputs offline and is structurally wired for live integration through production hooks.

## 2.2 The Alibaba SSD Dataset

### 2.2.1 Source and Structure

The Alibaba SSD Failure Dataset (Han et al., 2021) covers daily S.M.A.R.T. telemetry from over 500,000 enterprise SSDs deployed across Alibaba Cloud’s hyperscale data centers. The observation window spans January 1, 2018 to December 31, 2019 — 730 daily snapshots in total. Two file types are provided. First, 716 daily CSV files (`smartlog2018ssd.zip` and `smartlog2019ssd.zip`, totaling approximately 8.7 GB compressed), each containing all SMART readings collected on that calendar date. Each file has 105 columns: `disk_id`, `ds` (sampling date), `model` (architecture code), and 51 pairs of normalized ( $n_i$ ) and raw ( $r_i$ ) SMART counters. Second, `ssd_failure_label.csv` provides 16,305 confirmed drive failure records, each with the disk identifier, architecture code, and an exact failure timestamp. A validation check confirmed that the label file is clean: every `disk_id` appears exactly once, ruling out duplicate failure records.

### 2.2.2 Fleet Architecture Statistics

Six distinct SSD architectures are present, anonymized as MA1 through MC2. The MA and MB families use Multi-Level Cell (MLC) NAND flash; the MC family uses Triple-Level Cell (TLC) NAND. These differ in endurance, density, and failure behavior — and critically, in which S.M.A.R.T. attributes each controller firmware activates. Table 2.1 presents the key fleet statistics along with the final dataset sizes produced by the pipeline for each architecture.

| Architecture | NAND | Fleet Share | Failures Share | AFR (%) | SMART Features | Drive Count | Failures |
|--------------|------|-------------|----------------|---------|----------------|-------------|----------|
| MA1          | MLC  | 10.0%       | 20.9%          | 2.36%   | 25             | 50,000      | 3,408    |
| MA2          | MLC  | 25.7%       | 8.5%           | 0.46%   | 26             | 128,500     | 1,386    |
| MB1          | MLC  | 8.9%        | 15.7%          | 2.52%   | 31             | 44,500      | 2,560    |
| MB2          | MLC  | 10.4%       | 6.0%           | 0.71%   | 28             | 52,000      | 978      |
| MC1          | TLC  | 40.4%       | 37.8%          | 3.29%   | 29             | 202,000     | 6,163    |
| MC2          | TLC  | 4.6%        | 11.2%          | 3.92%   | 26             | 23,000      | 1,826    |

**Table 2.1:** Alibaba SSD Fleet Characteristics and Original Dataset Scale per Architecture.

The class imbalance varies substantially: MA2 has an AFR of only 0.46% — fewer than one in two hundred drives fails per year — while MC2 reaches 3.92%. MC1 alone accounts for 40.4% of the entire fleet and 37.8% of all failures, making it the dominant architecture in both scale and failure contribution. This heterogeneity prevents any uniform class-weighting strategy from being applied across architectures, which directly motivates the architecture-specialized Local Champion design explored in Chapter 3.

### 2.2.3 S.M.A.R.T. Attribute Heterogeneity

A critical discovery during the initial data exploration is that SMART attribute coverage is not uniform across architectures. Each manufacturer activates only the attributes relevant to its specific controller firmware. Table 2.2 maps the 19 SMART attributes considered in this study (following Xu et al., 2021) against the six architectures. The  $\star$  symbol in the "Universal" column marks the attributes present in all six, forming the cross-architecture intersection.

Only 11 attributes are present in every architecture (RSC, POH, PCC, PFC, EFC, MWI, ARS, ETE, UCE, temperature attributes, and CRC Error Count). Notably, the raw throughput counters  $r\_241$  (Total LBAs Written) and  $r\_242$  (Total LBAs Read) — among the most predictive features for MB1 — are absent in MA1, MB2, MC1, and MC2, which forces the universal models to rely on a more restricted feature set. This physical constraint shapes the universal dataset design described in Section 2.5.

| SMART Attribute                          | MA1 | MA2 | MB1 | MB2 | MC1 | MC2 | Universal |
|------------------------------------------|-----|-----|-----|-----|-----|-----|-----------|
| Reallocated Sectors Count (RSC / $r_5$ ) | ✓   | ✓   | ✓   | ✓   | ✓   | ✓   | ★         |
| Power-On Hours (POH / $r_9$ )            | ✓   | ✓   | ✓   | ✓   | ✓   | ✓   | ★         |
| Power Cycle Count (PCC / $r_{12}$ )      | ✓   | ✓   | ✓   | ✓   | ✓   | ✓   | ★         |
| Program Fail Count (PFC)                 | ✓   | ✓   | ✓   | ✓   | ✓   | ✓   | ★         |
| Erase Fail Count (EFC)                   | ✓   | ✓   | ✓   | ✓   | ✓   | ✓   | ★         |
| Media Wearout Indicator (MWI)            | ✓   | ✓   | ✓   | ✓   | ✓   | ✓   | ★         |
| Available Reserved Space (ARS)           | ✓   | ✓   | ✓   | ✓   | ✓   | ✓   | ★         |
| End-to-End Error (ETE)                   | ✓   | ✓   | ✓   | ✓   | ✓   | ✓   | ★         |
| Uncorrectable Errors (UCE / $r_{187}$ )  | ✓   | ✓   | ✓   | ✓   | ✓   | ✓   | ★         |
| Temp: Enclosure (ET) & Airflow (AFT)     | ✓   | ✓   | ✓   | ✓   | ✓   | ✓   | ★         |
| CRC Error Count (CEC / $r_{199}$ )       | ✓   | ✓   | ✓   | ✓   | ✓   | ✓   | ★         |
| Wear Range Delta ( $r_{177}$ )           | ×   | ×   | ✓   | ✓   | ×   | ×   | —         |
| Total LBAs Written ( $r_{241}$ )         | ×   | ✓   | ✓   | ×   | ×   | ×   | —         |
| Total LBAs Read ( $r_{242}$ )            | ×   | ✓   | ✓   | ×   | ×   | ×   | —         |
| Power Loss Protection Failure (PLP)      | ✓   | ✓   | ×   | ×   | ×   | ×   | —         |
| Unexpected Power Loss Count (UPL)        | ✓   | ✓   | ×   | ×   | ✓   | ✓   | —         |
| Downshift Error Count (DEC)              | ×   | ✓   | ✓   | ✓   | ✓   | ✓   | —         |
| Command Timeout (CMDT)                   | ✓   | ×   | ×   | ×   | ✓   | ✓   | —         |
| Raw Read Error Rate (RER)                | ×   | ×   | ×   | ×   | ✓   | ✓   | —         |

**Table 2.2:** SMART Attribute Coverage Across Six Architectures.

## 2.3 The Three-Phase Production Pipeline

The full pipeline from raw archives to final unified datasets operates in three sequential phases. Figure 2.1 presents the complete architectural flow.

An important design principle of the pipeline is that all six model families — Random Forest, XGBoost, Attention-BiLSTM, and their ensemble — share exactly the same final input file per architecture (`{MODEL}_Ready_For_LSTM_Binary_V2.csv`). This guarantees a fair benchmark: any performance differences observed between models in Chapter 4 reflect the models’ learning capacity, not preprocessing artifacts. The intermediate file (`{MODEL}_final_ready_for_ML_40Seq_14D.csv`) retains natural temporal gaps and calendar-based labels for data auditing purposes but is not used directly for model training.

### 2.3.1 MB1 Baseline: Engineering the Pipeline

Before the production pipeline was designed, a systematic experimental campaign was conducted on the MB1 disk architecture exclusively. MB1 was selected as the baseline because it is a well-characterized Enterprise-class MLC drive (24 nm lithography, 1,920 GB capacity) with a moderate AFR of 2.52%, providing enough failure events for robust statistical evaluation while remaining tractable for iterative experimentation. The 1,807 confirmed failures in the MB1 label file span the entire 2018–2019 observation period with reasonable monthly distribution.

|                                                                                                                                                                                |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>RAW INPUT — Alibaba SSD Dataset</b>                                                                                                                                         |
| smartlog2018ssd.zip (4.33 GB, 361 daily CSV files) • smartlog2019ssd.zip<br>ssd_failure_label.csv (16,305 confirmed failure records)                                           |
| <b>PHASE 1 — Streaming Extraction &amp; Parquet Compression</b>                                                                                                                |
| Single-pass iteration over 716 daily CSV files • Per-model row filtering<br>Reservoir Filter (MC1/MA2) • strict float32 Parquet schema                                         |
| <b>6 × Per-Architecture Parquet Files (MA1 • MA2 • MB1 • MB2 • MC1 • MC2)</b>                                                                                                  |
| All SMART columns retained • Integer dates • float32 dtypes • RAM-safe schema                                                                                                  |
| <b>PHASE 2 — ML Sequence Building</b>                                                                                                                                          |
| Architecture Scanner (drop >99% NaN columns) → V3 DropNA → Ghost Data Removal<br>V6 Silent Failure Filter ( $\geq 7$ signal days) → Continuity + Right-Censoring → 1:1 Balance |
| <b>6 × Intermediate Datasets: {MODEL}_final_ready_for_ML_40Seq_14D.csv</b>                                                                                                     |
| Native SMART features (unscaled) • Natural temporal gaps in failed sequences                                                                                                   |
| <b>PHASE 3 — Final Unification (Calendar Healer + Scaling + Positional Labeling)</b>                                                                                           |
| Calendar Healer: linear interpolation of gaps → MinMaxScaler [0,1] globally<br>Positional binary labels (last 14 rows = 1) → 8-test validation suite                           |
| <b>6 × FINAL UNIFIED DATASETS: {MODEL}_Ready_For_LSTM_Binary_V2.csv</b>                                                                                                        |
| 40 continuous days • All features MinMax-scaled • Gap-free (interpolated)                                                                                                      |

**Figure 2.2:** Complete Three-Phase Data Engineering Pipeline.

The following sub-sections document each design decision in the pipeline and the experimental evidence that justified it.

## 2.4 Phase 1 — Streaming Extraction and Parquet Compression

Phase 1 processes the raw ZIP archives in a single streaming pass across all 716 daily CSV files. For each file, rows are partitioned by architecture and appended to six growing output files simultaneously. After all files are processed, each per-architecture CSV is re-read in 500,000-row chunks and converted to Apache Parquet format with a strict `float32` schema. Two large architectures required a Reservoir Filter to prevent RAM overflow: MA2 has 69 million rows and MC1 has 115 million rows in the raw archives, both exceeding the 16 GB development machine capacity. The reservoir filter pre-selects a bounded candidate pool of healthy disks (limited to  $3 \times$  the failed disk count) during streaming, discarding rows of excluded disks before they enter RAM. For MC1, this blocked 89,243,342 rows from reaching memory, reducing peak consumption from an estimated 45 GB to approximately 8 GB.

## 2.5 Phase 2 — ML Sequence Building and Constraint Validation

Phase 2 applies eight sequential constraints in-flight on each architecture’s Parquet file, producing the intermediate 40-day per-architecture dataset. The constraints were discovered iteratively during the MB1 baseline experiments. Table 2.3 summarizes the six versions of the MB1 dataset and the constraint each version introduced.

| Ver. | Key Constraint Added                  | What It Fixes        | Outcome            | Remaining Issue         |
|------|---------------------------------------|----------------------|--------------------|-------------------------|
| V1   | 5% downsampling; 60d window           | First assembly       | 222,107 rows       | 2,495 healthy with gaps |
| V2   | Candidate-pool: 60d continuity        | Eliminates gaps      | 1,807 F + 1,807 H  | 18 gaps remain (bug)    |
| V3   | DropNA BEFORE continuity              | Removes corrupted    | ALL 8 tests PASS   | 4 silent failures       |
| V4   | Silent Failure Audit                  | Removes mislabeled   | 1,803 F + 1,803 H  | 6 zero-variance cols    |
| V5   | Zero-Variance Filter (drop constants) | Removes placeholders | 36 active features | 15d window signal drop  |
| V6★  | 14d window · 7d floor · 40d           | Best F1              | 1,705 F + 1,705 H  | — (all issues resolved) |

**Table 2.3:** MB1 Dataset Pipeline Iteration History.

### 2.5.1 Architecture Scanner and V3 DropNA Rule

A null-rate analysis on the MB1 subset of the last daily file (2019-12-31, 38,564 rows) revealed that 63 of 105 columns maintained a 100% null rate for MB1 — corresponding to SMART

attributes not supported by the MB1 controller. These are eliminated automatically by the Architecture Scanner, which drops any column exceeding 99% nullity. After filtering, the remaining columns carry negligible missing rates ( $\sim 0.27\%$  for 2018,  $\sim 0.01\%$  for 2019). The V3 DropNA rule then removes the small fraction of rows with any remaining null value, applied before any downstream operation. The ordering is critical: V2 of the pipeline applied DropNA after the continuity check, which caused 18 healthy sequences to appear gap-free in the continuity audit but to acquire gaps after rows were removed by DropNA. Moving DropNA to first position eliminated this class of error entirely.

## 2.5.2 Ghost Data Elimination

"Ghost data" refers to SMART log rows for a disk that appear after the recorded failure date. Because the failure label file records the exact failure timestamp and the daily SMART logs capture the full day's state, a disk that failed at 09:06 AM on a given date would still appear in that day's log. More critically, logging infrastructure delays occasionally produced rows for replaced drives in subsequent days. The pipeline resolves this with a hard truncation rule: for every failed disk, all rows where `ds > failure_time` are unconditionally removed. A validation audit confirmed that after this rule, zero ghost rows exist in any final dataset.

## 2.5.3 The 14-Day Prediction Horizon

The prediction horizon  $H$  determines how many days before failure an observation receives label  $y = 1$ . Three candidate horizons were evaluated on MB1 using a 5-fold Random Forest experiment on the 40-day sequence dataset. Table 2.4 reports the results.

| Horizon $H$ | Avg Signal Days / Disk | Full Signal | RF Precision | RF Recall | Disk-Level F1 |
|-------------|------------------------|-------------|--------------|-----------|---------------|
| 7-day       | $\sim 5.8$             | 89.7%       | 0.6771       | 0.4241    | 0.5437        |
| 10-day      | $\sim 8.4$             | 84.2%       | 0.6413       | 0.4623    | 0.5403        |
| 14-day ★    | 13.2 (avg)             | 78.4%       | 0.6864       | 0.4775    | 0.5632 ★      |

**Table 2.4:** Prediction Horizon Comparison — Disk-Level Metrics for MB1.

The 14-day horizon yields the highest disk-level F1 (0.5632 at threshold 0.45). Although the 7-day window produces a denser signal, fourteen days provides the minimum reaction time a data center operator needs to procure a replacement drive and schedule a controlled migration without service interruption. All subsequent models use  $H = 14$  days. The target assignment rule is pure calendar math: a row receives  $y = 1$  if and only if the drive's official failure date is between 0 and 13 calendar days in the future.

## 2.5.4 Silent Failure Filter and Balance Constraints

A "silent failure" is a disk whose confirmed failure falls entirely outside the 40-day observation window — it died before the window starts, or its last log was recorded more than 40 days before failure. Such a disk has all 40 rows labeled  $y = 0$ , yet it belongs to the failed-disk pool. The V6 pipeline applies a 7-day minimum signal floor: any failed disk with fewer than 7 positive-label rows is excluded. This removed 98 disks from the MB1 pool. The rationale is straightforward: a disk with only 1–2 positive days provides contradictory supervision — 38 out of 40 rows signal health while the disk is actually failing. The dataset is balanced at the disk level: the number of retained failed disks equals the number of retained healthy disks. For healthy disks, the right-censoring constraint prevents any disk whose last SMART log falls on or after December 1, 2019 from being included — such disks may have failed in January 2020, and treating them as permanently healthy introduces survivorship bias. The vectorized continuity check confirms that each selected healthy disk possesses at least one uninterrupted 40-calendar-day block before the cutoff date.

## 2.6 Temporal Labeling and Sequence Construction

### 2.6.1 The 40-Day Sequence Window

The sequence length  $T = 40$  days was selected based on the continuity properties of the SMART logging infrastructure. Table 2.5 characterizes the gap distribution across all six architectures for the failed disk pool only (healthy sequences, by construction, are verified gap-free before selection).

| Arch. | Total Failed | Disks with Gaps | Gap Rate (%) | Max Gap  | Implication                                |
|-------|--------------|-----------------|--------------|----------|--------------------------------------------|
| MA1   | 968          | 508             | 50.9%        | 147 days | Moderate interpolation required            |
| MA2   | 486          | 295             | 58.5%        | 99 days  | Moderate interpolation required            |
| MB1   | 1,667        | 1,101           | 64.8%        | 126 days | High – most critical architecture          |
| MB2   | 565          | 284             | 49.6%        | 575 days | Extreme instability before failure         |
| MC1   | 10,015       | 4,965           | 48.9%        | 437 days | Large scale – vectorized healing essential |
| MC2   | 996          | 653             | 61.8%        | 253 days | Significant pre-failure instability        |

**Table 2.5:** Temporal Gap Distribution in Failed Disk Sequences per Architecture.

Between 49% (MB2) and 65% (MB1) of failing drives have at least one missing logging day within their final 40-day window. These gaps reflect hardware instability in the final days of life — a disk experiencing I/O errors may fail to submit its daily SMART report. Extending the window beyond 40 days would incorporate larger gaps (MB2 shows a maximum gap of 575 days), where linear interpolation over hundreds of synthetic days would introduce more artificial signal than real sensor data. The 40-day window captures the critical degradation phase while keeping the interpolation burden tractable.

## 2.6.2 Binary Labeling: Calendar vs. Positional

The intermediate dataset uses calendar-based labeling (Phase 2): a row receives  $y = 1$  if the drive's failure date is  $\leq 13$  calendar days away. This is exact by construction but produces variable numbers of positive rows per disk (between 7 and 14) because failing disks may skip logging days. The final unified dataset uses positional labeling (Phase 3, after the Calendar Healer): since all sequences are exactly 40 continuous calendar days after gap-filling, the last 14 rows always correspond to exactly the last 14 calendar days. The positional rule (last 14 rows = 1, first 26 rows = 0) is therefore identical to the calendar rule — but only valid after the Calendar Healer has guaranteed temporal continuity.

## 2.7 Phase 3 — Final Unification: Calendar Healer, Scaling, and Positional Labeling

Phase 3 transforms each architecture's intermediate dataset into the final unified dataset (`{MODEL}_Ready_For_LSTM_Binary_V2.csv`) shared by all model families. Three operations are applied in strict order: **Calendar Healer**. Each disk's date range is expanded to a full calendar sequence from its first to its last log day. Missing dates are inserted as blank rows, and their SMART sensor values are filled using linear interpolation, with forward-fill and backward-fill applied at boundary gaps. After healing, every disk contributes exactly 40 gap-free consecutive days. The calendar healer is the step that enables all model families — including tree-based models that process rows independently — to operate on the same clean dataset, and that makes positional labeling exact. **MinMax Scaling**. All SMART features are scaled to the  $[0, 1]$  range using a MinMaxScaler fitted on the combined training set for each architecture. This is necessary for the Attention-BiLSTM (which is sensitive to feature magnitude) and harmless for tree-based models (which are scale-invariant). Scaling after the Calendar Healer ensures that interpolated values are scaled consistently with real observations. **Positional Binary Labeling**. For each failed disk, rows 27–40 (the last 14 days of the 40-day sequence) receive `y_class = 1`; rows 1–26 receive `y_class = 0`. All healthy disk rows receive `y_class = 0`. A post-processing audit of the MB1 final dataset confirmed that 100% of the 1,667 failed disks have exactly 14 positive labels with zero time gaps inside the failure window.

### 2.7.1 Key SMART Features and Predictive Value

During the MB1 experimental campaign, a multi-metric feature ranking was computed combining Pearson correlation, Spearman correlation, Youden's J-index (single-feature threshold performance), Random Forest importance, and XGBoost importance. Table 2.6 presents the

top-ranked features for MB1, alongside their physical interpretation.

| Rank | SMART Attribute                        | SHAP (XGB) | RF Imp.       | Cons. Score | Physical Meaning               |
|------|----------------------------------------|------------|---------------|-------------|--------------------------------|
| 1    | $n_5 / r_5$ – Reallocated Sector Count | High       | 0.030 / 0.097 | 0.624       | Bad-block retirements          |
| 2    | $r_9$ – Power-On Hours (raw)           | Highest    | 0.220         | 0.622       | Total operational age          |
| 3    | $r_5$ – RSC raw value                  | High       | 0.097         | 0.495       | Distinct raw signal from $n_5$ |
| 4    | $r_{241}$ – Total LBAs Written         | High       | 0.152         | 0.488       | Cumulative write workload      |
| 5    | $r_{183}$ – Runtime Bad Blocks         | High       | 0.000         | 0.486       | Late-life degradation signal   |
| 6    | $r_{177}$ – Wear Range Delta           | Medium     | 0.097         | 0.466       | Wear uniformity signal         |

**Table 2.6:** Top-6 SMART Features for MB1 by Consolidated Multi-Metric Ranking.

Power-On Hours ( $r_9$ ) emerges as the dominant predictor — a finding consistent with the field-study literature (Meza et al., 2015; Xu et al., 2021), which shows that failure risk increases non-linearly with operational age. Reallocated Sector Count ( $n_5/r_5$ ) is the second most informative attribute, reflecting the progressive accumulation of bad blocks as NAND cells exhaust their P/E cycle budget. The throughput counters  $r_{241}$  and  $r_{242}$ , exclusive to MB1 and MA2, capture the cumulative write workload and serve as a proxy for flash wear. These six attributes collectively explain the majority of the model’s predictive signal for MB1, while 73% of MB1’s failed disks exhibit interpretable degradation trajectories with at least 7 days of rising risk probability before failure.

## 2.8 Validation Framework

Every final dataset produced by the pipeline is subjected to an automated suite of eight validation tests before being used in any modeling experiment. The suite was developed during the MB1 experimental phase and applied identically to all six architecture-specific datasets and all universal datasets. Table 2.7 describes each test, the failure it guards against, and the collective result.

| #  | Constraint Validated          | What Failure Means                | Result (All 6 Datasets)              |
|----|-------------------------------|-----------------------------------|--------------------------------------|
| T1 | Zero missing values           | NaN in input → undefined gradient | PASS — 0 NaN values                  |
| T2 | No data leakage               | Disk appears in both classes      | PASS — zero ID overlap               |
| T3 | Exact 1:1 disk-level balance  | Skewed class distribution         | PASS — symmetric failure rates       |
| T4 | All sequences exactly 40 days | Variable-length sequences         | PASS — all disks: exactly 40 rows    |
| T5 | 100% calendar continuity      | Gaps in healthy seqs              | PASS — zero time gaps                |
| T6 | Ghost data elimination        | Post-failure rows mislabeled      | PASS — 0 ghost rows detected         |
| T7 | 14-day target window          | Positional labeling over gaps     | PASS — window mathematically perfect |
| T8 | Right-censoring cutoff        | Healthy may have failed later     | PASS — 0 sequences enter Dec 2019    |

**Table 2.7:** Eight-Test Validation Suite Results.

All six architecture-specific datasets and all cross-architecture universal datasets passed every test before being used for training. This systematic validation is what makes the performance benchmarks of Chapter 4 scientifically reliable: any observed F1 difference between models reflects genuine signal differences in the SMART data, not preprocessing inconsistencies.

## 2.9 Universal Dataset Construction

Beyond the six architecture-specific datasets, three families of cross-architecture datasets were produced for the Universal Blind and Hardware-Aware modeling experiments. These datasets are assembled from the six final unified datasets.

### 2.9.1 Feature Intersection Analysis

A header-scan across all six final datasets identified the set of features shared across every architecture. The result is a six-feature universal intersection: *n\_5* (normalized RSC), *r\_5* (raw RSC), *r\_9* (Power-On Hours), *r\_12* (Power Cycle Count), *r\_187* (Reported Uncorrectable Errors), and *r\_199* (CRC Error Count). This minimal intersection is a direct consequence of Table 2.2: the high-information-value throughput counters *r\_241*/*r\_242* are absent in four of the six architectures, and the PLP and UPL attributes are architecture-specific. The Universal Blind hypothesis is therefore tested under the most restrictive possible feature constraint: five native SMART sensors available on every drive family in the fleet. Any performance gap between Universal Blind and Local Champions is thus attributable to both the loss of architecture-specific features and the loss of specialization — making the near-parity result demonstrated in Chapter 4 particularly meaningful.

### 2.9.2 Universal\_SSD\_Dataset\_V1 (Balanced, 5-Feature)

The `Universal_SSD_Dataset_V1.csv` is assembled from the six final unified datasets by: (1) prefixing each `disk_id` with the architecture code (e.g., `MA1_12345`) to prevent ID collisions; (2) retaining only the five common features plus the `y_class` target; (3) applying balanced cross-architecture subsampling — exactly 486 failed and 486 healthy disks are drawn from each architecture using a fixed random seed, matching the bottleneck imposed by MA2 (the smallest architecture, with 486 qualified failed disks). The result is 5,832 unique drives, 2,916 failed and 2,916 healthy, and 233,280 total rows. This dataset is the exclusive training corpus for the Universal Blind RF and Universal Blind XGBoost models.

### 2.9.3 Hardware-Aware Datasets (Sparse Union with One-Hot Encoding)

The Hardware-Aware datasets test a third hypothesis: does injecting explicit architecture identity into the feature vector allow a single model to match the Local Champions while maintaining the MLOps simplicity of the Universal Blind approach? These datasets are constructed by concatenating all six final unified datasets into a single sparse union matrix. Columns absent from a given architecture appear as NaN for that architecture's rows. Six binary indicator

features (`is_Model_MA1`, `is_Model_MA2`, ..., `is_Model_MC2`) encode the architecture identity via one-hot encoding. Two variants were produced. The balanced version (990 disks per architecture, stratified) is used for the Hardware-Aware XGBoost and Hardware-Aware Attention-BiLSTM. The full unbalanced version (29,675 disks total, 40.29% matrix sparsity) maximizes training data but introduces cross-architecture class imbalance. XGBoost handles sparse matrices natively via its learned branching mechanism for NaN. The BiLSTM-specific variant replaces all NaN values with  $-999.0$  — a sentinel value lying far outside the  $[0, 1]$  MinMax range, which the network learns to interpret as a "feature absent" signal.

## 2.9.4 Universal Dataset Audit Results

All universal cross-architecture datasets were subjected to the same automated validation suite as the per-architecture unified datasets. Tables 2.9 and 2.10 report the complete audit results for the two primary Hardware-Aware datasets: the balanced configuration (5,940 disks, 990 per architecture) and the full unbalanced configuration (29,675 disks, natural fleet distribution). Both datasets passed all four tests without exception, confirming that the sparse-union assembly procedure preserves sequence integrity and class stratification across all six architectures.

| Test | Constraint           | Result                                                       |
|------|----------------------|--------------------------------------------------------------|
| T1   | Sequence Integrity   | [PASS] Every hard drive has exactly 40 chronological days.   |
| T2   | Cross-Arch Balance   | [PASS] Perfect Balance: All 6 architectures have 990 drives. |
| T3   | Class Stratification | [PASS] GLOBAL: 5,940 disks   Healthy: 3,012   Failed: 2,928  |
| T4   | Column Density Check | [PASS] No NaN values in Target, disk_id, date, or One-Hot.   |

**Table 2.8:** Audit Results — `Balanced_Dataset_XGBoost_Hardware_Aware.csv`.

| Test | Constraint          | Result                                                         |
|------|---------------------|----------------------------------------------------------------|
| T1   | Sequence Integrity  | [PASS] Every hard drive has exactly 40 chronological days.     |
| T2   | Volume Verification | [PASS] Expected 29,675 disks — Actual 29,675 disks.            |
| T3   | Class Distribution  | [PASS] GLOBAL: 29,675 disks   Healthy: 14,978   Failed: 14,697 |

**Table 2.9:** Audit Results — `Ultimate_Dataset_XGBoost_Hardware_Aware.csv`.

## 2.10 Final Dataset Statistics

Table 2.8 provides a complete inventory of all datasets produced by the pipeline, with their key statistics and the model families that consume them.

The total modeling footprint of the dataset collection is substantial: 29,675 unique physical drives observed over 40-day windows, representing over 1.1 million daily SMART snapshots for the largest single dataset. The per-architecture dataset sizes range from 990 disks (MA2) to

| Dataset File             | Scope            | Disks        | Rows              | Features           | Consumer Model(s)      |
|--------------------------|------------------|--------------|-------------------|--------------------|------------------------|
| {MODEL}_Ready_V2.csv     | Per-architecture | Proportional | Prop. $\times$ 40 | Native per arch    | RF · XGB · BiLSTM      |
| Universal_SSD_Dataset    | All 6 balanced   | 5,832        | 233,280           | 5 universal        | Universal Blind RF/XGB |
| Balanced_XGB_HW_Aware    | All 6 balanced   | 5,940        | 237,600           | $\sim$ 57 features | HW-Aware XGBoost       |
| Ultimate_XGB_HW_Aware    | All 6 full union | 29,675       | 1,187,000         | $\sim$ 57 sparse   | Large-scale HW XGBoost |
| Balanced_BiLSTM_HW_Aware | All 6 imputed    | 5,940        | 237,600           | $\sim$ 57 imputed  | HW-Aware BiLSTM        |

**Table 2.10:** Complete Dataset Inventory.

20,163 disks (MC1), reflecting the natural fleet distribution. For all datasets, the failure rate at the disk level is approximately 49%, with the exact value ranging from 48.5% (MC2) to 49.7% (MC1/MB2) due to rounding from the 1:1 balance constraint.

## 2.11 Modelling: The Local Champions Suite

The local champions approach trains a dedicated model per drive architecture and selects the best-performing variant per architecture. The portfolio comprises four families: Random Forest, XGBoost, a causal Attention-BiLSTM, and a Supreme Soft-Voting Ensemble. Each architecture is treated as a self-contained development cycle, with the methodology established on the MB1 reference architecture and then re-applied identically to MA1, MA2, MB2, MC1, and MC2 by swapping the input dataset only.

### 2.11.1 The Random Forest Champion

The Random Forest implementation uses scikit-learn’s `RandomForestClassifier` with class-weighted training (`class_weight='balanced_subsample'`). Hyperparameter selection follows a two-stage protocol: a wide `RandomizedSearchCV` over a precision-focused F0.5 scorer, followed by a 5-fold `StratifiedGroupKFold` cross-validation of the leading candidate. The F0.5-optimal configuration on the MB1 data is reported in Table 2.11.

**Table 2.11:** Random Forest hyperparameters from the F0.5-optimized randomised search on MB1.

| Hyperparameter                 | Value                |
|--------------------------------|----------------------|
| <code>n_estimators</code>      | 300                  |
| <code>max_depth</code>         | 14                   |
| <code>min_samples_split</code> | 10                   |
| <code>min_samples_leaf</code>  | 5                    |
| <code>max_features</code>      | "sqrt"               |
| <code>class_weight</code>      | "balanced_subsample" |
| <code>random_state</code>      | 42                   |

The cross-fold training proceeds under 5-fold `StratifiedGroupKFold` using `disk_id` as the

group identifier, so that every observation from a given drive falls entirely within one fold. The model is fit on four of the five folds, and the predicted probabilities on the held-out fold are written to an Out-of-Fold (OOF) predictions array. After all five folds are completed, every row in the dataset carries an OOF probability that was produced by a model that did not see that row's drive during training.

The Random Forest probability vector is then reshaped into the (Disks, 40) tensor and passed through a causal 3-day uniform smoothing filter to absorb single-day spikes. The maximum smoothed probability per drive becomes the disk-level prediction.

## 2.11.2 The XGBoost Champion

The XGBoost implementation uses the official `xgboost` Python package with the `binary:logistic` objective and `tree_method='hist'`. The F0.5-optimal configuration on the MB1 data is reported in Table 2.12.

**Table 2.12:** XGBoost champion hyperparameters from the F0.5-optimised randomised search on MB1.

| Hyperparameter                | Value             |
|-------------------------------|-------------------|
| <code>n_estimators</code>     | 400               |
| <code>max_depth</code>        | 6                 |
| <code>min_child_weight</code> | 5                 |
| <code>learning_rate</code>    | 0.05              |
| <code>gamma</code>            | 0.1               |
| <code>reg_lambda</code>       | 2.0               |
| <code>reg_alpha</code>        | 0.5               |
| <code>subsample</code>        | 0.8               |
| <code>colsample_bytree</code> | 0.8               |
| <code>scale_pos_weight</code> | per-fold computed |
| <code>tree_method</code>      | "hist"            |

The same training pipeline used for the Random Forest is reused identically with XGBoost. The two models differ in two important respects on this corpus. First, XGBoost is roughly four times faster to train per fold (because the histogram-based split-finding is more efficient). Second, XGBoost's *sparsity-aware* split-finding is the engine that makes the Universal Blind model of Section 2.12 viable; on the per-architecture champion, the feature set is fully populated (no missingness), so the sparsity-aware machinery is dormant.

### 2.11.3 The Causal Attention-BiLSTM Champion

The deep-learning champion is a *causal* Attention-BiLSTM, designed deliberately to be deployable in a streaming day-by-day inference pipeline without any leakage from future timesteps.

The architectural innovation that makes a bidirectional LSTM causal is the **temporal unroller**. Given a 40-day input sequence, the unroller produces 40 different sequences, where the  $t$ -th unrolled sequence contains only days 1 through  $t$  (with days  $t + 1$  through 40 masked to a learnable pad token). The BiLSTM is then trained on the unrolled tensor in a many-to-one configuration. At inference time, the unroller is applied once at the start of each day’s observation, and the inference cost is the same as a unidirectional LSTM of equivalent depth.

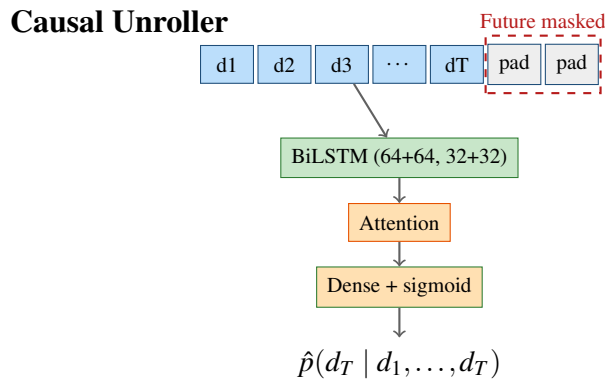


**Figure 2.3:** Causal Attention-BiLSTM architecture. The temporal unroller pre-masks every future timestep before the BiLSTM sees the sequence, so that the bidirectional connection is exploited at training time but the model produces day- $T$  predictions using only days 1 through  $T$  at both training and inference time.

The full architecture is: input tensor of shape (batch, 40, 8); causal unroller; BiLSTM layer 1 with 64 units in each direction and dropout 0.2; BiLSTM layer 2 with 32 units in each direction and dropout 0.2; Bahdanau-style additive attention layer with a learnable query vector; dense layer with 32 units and ReLU activation and L2 regularisation  $10^{-3}$ ; and a single sigmoid output unit.

The model is trained with Adam (`learning_rate` =  $10^{-3}$ , `clipnorm` = 1.0), binary cross-entropy loss with sample weighting set to  $2.4\times$  for positive-label days, and dynamic-temporal weighting in which the positive-day weight ramps linearly from 1.0 at day 1 of the 40-day window to 2.4 at day 40. Training runs for 60 epochs with `EarlyStopping` (`patience` = 10, `restore_best_weights` = True) and `ReduceLROnPlateau` (`factor` = 0.5, `patience` = 4).

Several intermediate architectures were evaluated on the MB1 development data before the causal Attention-BiLSTM was selected as the deep-learning champion. The progression is summarised in Table 2.13.

**Table 2.13:** Architectural progression of the deep-learning champion on the MB1 development data. Each row is a separate training run; the best disk-level F1 across the threshold sweep is reported.

| Variant                            | Best F1       | Best Threshold | Notes                          |
|------------------------------------|---------------|----------------|--------------------------------|
| Baseline LSTM (many-to-many)       | 0.7567        | 0.40           | Starting point                 |
| LSTM + sample weighting            | 0.7515        | 0.70           | Precision shift                |
| LSTM + dynamic temporal weighting  | 0.7505        | 0.60           | Ramp-up confirmed              |
| LSTM + 5-fold ensemble + smoothing | 0.7673        | 0.60           | First above 0.75               |
| LSTM + LayerNorm + ElasticNet      | 0.7432        | 0.50           | Overfit narrows                |
| Causal BiLSTM                      | 0.7943        | 0.45           | ~2 F1 pts gain                 |
| CNN-LSTM                           | 0.7771        | 0.45           |                                |
| Patch-TST Transformer              | 0.7343        | 0.40           |                                |
| 40-Day Transformer                 | 0.7223        | 0.45           |                                |
| iTransformer                       | 0.6762        | degen.         | Failed to learn                |
| TSH-Transformer                    | 0.7302        | 0.40           |                                |
| <b>Causal Attention-BiLSTM</b>     | <b>0.7977</b> | <b>0.40</b>    | <b>Champion (lead 13.30 d)</b> |

The Transformer variants underperformed the recurrent ones on this corpus, consistent with the observation that Transformer-based methods require substantially larger labelled failure corpora to outperform recurrent baselines.

### 2.11.4 The Supreme Soft-Voting Ensemble

The fourth member of the local-champion portfolio is the Supreme Ensemble, a soft-voting combination of the Random Forest, XGBoost, and Attention-BiLSTM with uniform weights. The combination operates on the probability vectors, not the hard decisions: the ensemble probability is the per-day arithmetic mean of the three members' probabilities, smoothed by the standard 3-day causal filter and aggregated by the disk-level max.

The motivation is that the three members have systematically different error modes. The Random Forest is conservative and tends to favour precision over recall; the XGBoost is comparably conservative but with slightly different decision boundaries; the Attention-BiLSTM trades a small loss in precision for a substantial gain in recall through its causal temporal modelling. Averaging the three probabilities reduces the variance of the combined prediction and typically improves the disk-level F1 by 0.5–1.5 points over the best individual member.

## 2.12 Modelling: The Universal Hypothesis

The local-champions portfolio delivers four models per architecture, which means twenty-four model artefacts across six architectures. The operational implication is that every change in the pipeline must propagate through twenty-four separate training runs, twenty-four monitoring dashboards, twenty-four governance reviews. The Universal Hypothesis tests whether this multiplicity can be collapsed.

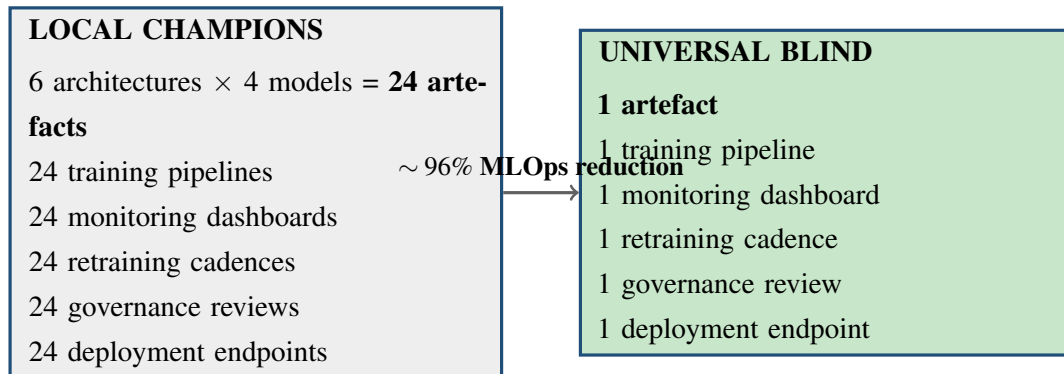


**Figure 2.4:** The Local Champions vs Universal Blind philosophical comparison. The Universal approach is operationally attractive only if its predictive performance is close enough to the per-architecture champions that the MLOps savings justify the F1 loss.

### 2.12.1 The Common-Features Baseline

The simplest universal model trains a single XGBoost on the intersection of S.M.A.R.T. attributes that all six architectures report. The strict intersection, after constant-input elimination, contains only five strongly-predictive features. This Common-5 baseline is implemented identically to the per-architecture champions but trained on the union of all six architectures. Chapter 3 reports the disk-level F1 of 0.7128 at threshold 0.45 — significantly weaker than every per-architecture champion. The Common-5 baseline is therefore a *negative result*: feature intersection is insufficient to support a single hardware-agnostic model.

### 2.12.2 The Blind vs Hardware-Aware Sparsity-Aware Variants

The second universal attempt exploits the sparsity-aware split-finding algorithm of XGBoost [3] directly. Instead of restricting the feature set to the intersection, the model is presented with the *union* of all S.M.A.R.T. attributes that any architecture reports — typically fifty-four features after per-architecture feature engineering. For each drive, the features that its architecture does not report are populated with NaN. At each candidate split, XGBoost enumerates only the non-missing values and learns the optimal default direction for missing values from the data —

effectively allowing the model to route around the architectural feature differences without ever being told the architecture identity.

| n_5  | r_9  | r_241 | r_177 | r_5  | r_190 | ... | r_198 |             |
|------|------|-------|-------|------|-------|-----|-------|-------------|
| 0.95 | 0.67 | NaN   | 0.42  | NaN  | 0.50  | ... | NaN   | ← MA1 drive |
| 0.88 | 0.55 | 0.30  | NaN   | NaN  | 0.45  | ... | NaN   | ← MB1 drive |
| 0.92 | 0.71 | NaN   | NaN   | 0.18 | 0.48  | ... | 0.05  | ← MC1 drive |

*Sparsity-aware split-finder: at each candidate node, enumerate only the non-NaN values of each feature; learn a default direction for NaNs from the training data.*

**Figure 2.5:** Sparsity-aware tree split on the heterogeneous S.M.A.R.T. feature vector. Each architecture populates a different subset of the 54-dimensional feature space; the empty cells are filled with NaN and the XGBoost split-finder routes them implicitly through the learned default direction.

To test whether the implicit routing matches what an explicit architecture label would buy, the same model is trained in two variants: a **Universal Blind XGBoost** trained on the 54-feature heterogeneous vector with no architecture labels, and a **Universal Hardware-Aware XGBoost** trained on the same vector plus a 6-dimensional one-hot encoding of the drive’s architecture. Both variants are also trained twice — once on the natural “Unbalanced” 29,675-disk corpus, once on a “Balanced” 5,940-disk subset. Table 2.14 summarises the seven universal-model configurations evaluated in Chapter 3.

**Table 2.14:** Universal model variants and their feature configurations.

| Variant                                | Drives        | Features                 | Arch. Label    | Deep?     |
|----------------------------------------|---------------|--------------------------|----------------|-----------|
| Common-5 RF                            | 5,832         | 5                        | None           | No        |
| Common-5 XGBoost                       | 5,832         | 5                        | None           | No        |
| Common-5 BiLSTM                        | 5,832         | 5                        | None           | Yes       |
| Universal Blind (Balanced)             | 5,940         | 54 (sparse)              | None           | No        |
| Universal HW-Aware (Balanced)          | 5,940         | 60 (sparse + OHE)        | One-hot        | No        |
| Universal Blind (Unbalanced)           | 29,675        | 54 (sparse)              | None           | No        |
| <b>Universal HW-Aware (Unbalanced)</b> | <b>29,675</b> | <b>60 (sparse + OHE)</b> | <b>One-hot</b> | <b>No</b> |

## 2.13 Evaluation Methodology

### 2.13.1 5-Fold StratifiedGroupKFold Cross-Validation

Every model in Smart Storage AI is trained under a 5-fold StratifiedGroupKFold protocol. The group identifier is the `disk_id`, so that every observation from a given drive falls entirely within

one fold; the stratification is on the disk-level binary failure label, so that every fold preserves the 1:1 class balance. The protocol prevents the two principal forms of leakage: temporal leakage (a drive’s observations appearing in both train and test) and class-imbalance leakage.

### 2.13.2 Disk-Level Smoothed Evaluation

The decisive choice in the evaluation protocol is that the primary metrics are *disk-level*, not cell-level. The argument is operational: the consequence of an alarm in a production AIOps system is a ticket per drive, regardless of how many days of the drive’s 40-day prediction window the model lit up. Computing precision and recall at the cell level is academically convenient but operationally misleading.

The exact disk-level computation is:

```
smoothed_probs = uniform_filter1d(raw_probs, size = 3, axis = 1, mode = nearest)
disk_max_prob = smoothed_probs.max(axis = 1)
disk_truth = (y_seq_true.sum(axis = 1) > 0)
disk_pred = (disk_max_prob ≥ threshold)
```

The lead time is computed by finding, for each true positive, the first day in the 40-day sequence at which the smoothed probability crossed the threshold:

$$\text{lead\_time} = 40 - \text{argmax}(\text{smoothed\_probs} > \text{threshold})$$

**Table 2.15:** The strict 1:1 disk-level metric definitions used throughout Smart Storage AI.

| Metric               | Definition                                                                 |
|----------------------|----------------------------------------------------------------------------|
| Disk-Level Precision | $\text{TP} / (\text{TP} + \text{FP})$ , at disk level                      |
| Disk-Level Recall    | $\text{TP} / (\text{TP} + \text{FN})$ , at disk level                      |
| Disk-Level F1        | $2 \cdot P \cdot R / (P + R)$                                              |
| Disk-Level F0.5      | $1.25 \cdot P \cdot R / (0.25 \cdot P + R)$ — precision weighted $2\times$ |
| Catch Rate           | = Disk-Level Recall                                                        |
| False Alarm Rate     | $\text{FP} / (\text{FP} + \text{TN})$ , on healthy-disk subset             |
| Average Lead Time    | Mean of lead_time over the TP set, in days                                 |

### 2.13.3 Causal Smoothing and the Blind Test Vault

The 3-day uniform causal filter applied before disk-level aggregation absorbs single-day prediction spikes. The filter is causal in the sense that the smoothed value at day  $t$  depends only on

the predictions at days  $t - 2, t - 1, t$ . Wider windows (5, 7 days) were evaluated and produced uniformly worse F1.

For the dashboard demo, Smart Storage AI carves a 15% blind test vault of 4,453 drives from the full 29,675-disk universal corpus, stratified by both architecture and class. This vault is the dataset replayed by the dashboard’s Live Simulation page; the predictions are pre-computed by the Universal Blind XGBoost and the four Local Champions per architecture.

## 2.14 The Smart Storage AI Mission-Control Dashboard

A predictive-maintenance pipeline that ends at a Kaggle notebook is operationally inert. The Smart Storage AI Dashboard is a five-page Streamlit mission-control platform running all five competing model families simultaneously on a 4,453-drive blind vault, designed to settle the central MLOps question: *do we need a model per architecture, or can a single Universal model do the job?*

### 2.14.1 Architecture and Design Philosophy

The dashboard is a *visualisation layer* over pre-computed prediction tensors. There is no live inference, no model serving, no websocket feed. Two data files are loaded once at startup: `Aligned_Master_Fleet.pkl` (4,453-drive blind vault, five models’ 40-day probability arrays) and `Dashboard_Live_Fleet.pkl` (100 drives: 70 healthy, 30 failing). Streamlit’s `@st.cache_resource` caches both loaders for instant page navigation. At the  $\sim 1$  Hz tick rate of the Live Simulation page, Streamlit’s `streamlit-autorefresh` is more than sufficient, and the dark Bloomberg-terminal theme is available for free.

The five pages are structured around two operator personas: the **NOC Operator** (needs one number per drive, one action per alert) and the **Storage Engineer** (needs all five models simultaneously, the full 40-day trajectory, and the A/B comparison). Every page supports both modes.

## 2.14.2 Mission Control Page

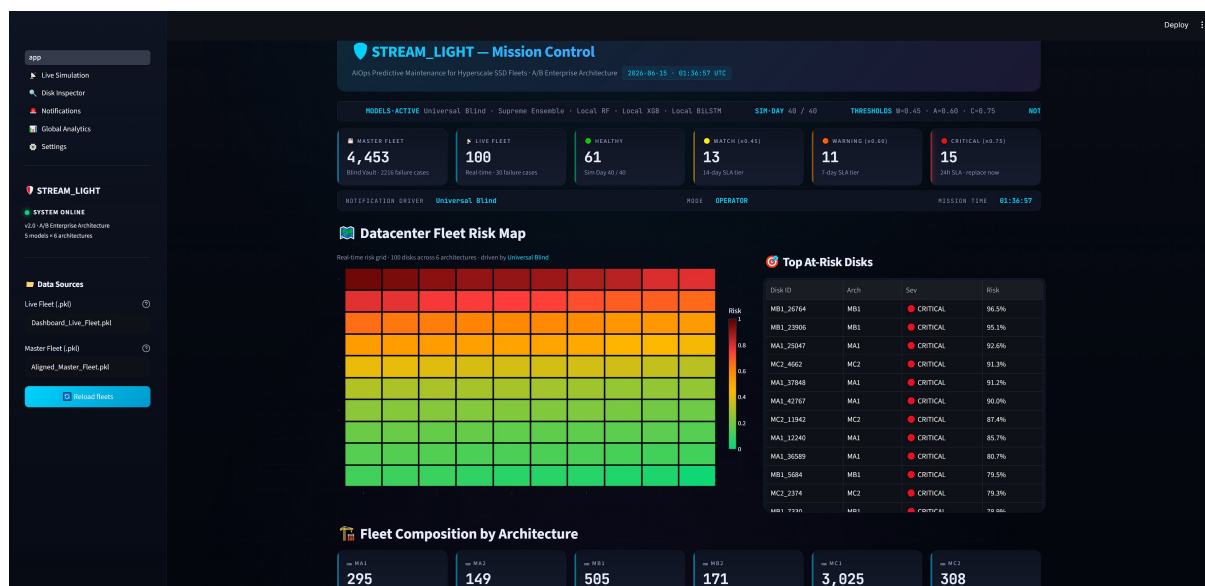


**Figure 2.6:** Smart Storage AI Dashboard — Mission Control page. Six KPI tier cards display Master Fleet (4,453 drives), Live Fleet (100 drives), Healthy (61), Watch (13), Warning (11), and Critical (15). The Fleet Risk Map heatmap colour-codes all 100 live drives by Universal Blind risk score. The Top At-Risk Disks table ranks drives by probability.

The Mission Control landing page (Figure 2.6) provides the real-time situational awareness layer. An animated ticker scrolls fleet statistics: *Master Fleet: 4,453 drives · Live Fleet: 100 drives · 5 Models · 6 Architectures*. At Day 40/40 of the simulation the live fleet reads 61 Healthy · 13 Watch · 11 Warning · 15 Critical. The Fleet Risk Map is a colour-coded heatmap where each cell represents one drive; cell colour maps to the Universal Blind risk score on a green-yellow-red scale. The Top-At-Risk Disks table ranks all flagged drives by risk score with Severity, Architecture, Forecast, and Truth columns. At the bottom, six architecture inventory cards show the fleet breakdown per disk family (MA1:295 · MA2:149 · MB1:505 · MB2:171 · MC1:3,025 · MC2:308).

## 2.14.3 Live Simulation Page

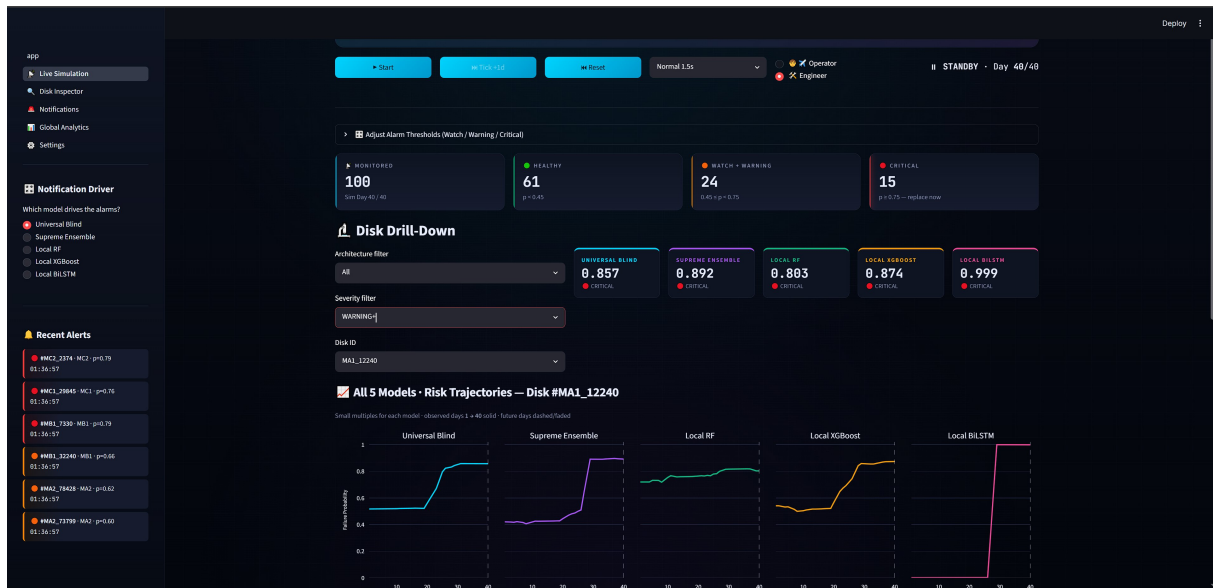


**Figure 2.7:** Smart Storage AI Dashboard — Live Simulation in Engineer mode. The 5-panel small-multiples grid shows per-model risk trajectories for a selected drive. The overlay chart displays all five model probability curves simultaneously. The Fleet State table ranks all live drives by current severity.

The Live Simulation page (Figure 2.7) replays predictions day-by-day at  $\sim 1.5$  s per day. **Operator mode** shows only the Universal Blind risk score — one number, one trajectory chart, the NOC view. **Engineer mode** expands to five stacked KPI cards (one per model in model-specific colours), a 5-panel small-multiples trajectory grid, and a single overlay chart with all five model curves. The Adjust Alarm Thresholds expander allows live tuning of Watch, Warning, and Critical probability cutoffs; the entire live fleet reclassifies immediately. The sidebar **Notification Driver** radio selector controls which model drives the alert feed.

## 2.14.4 Disk Inspector Page

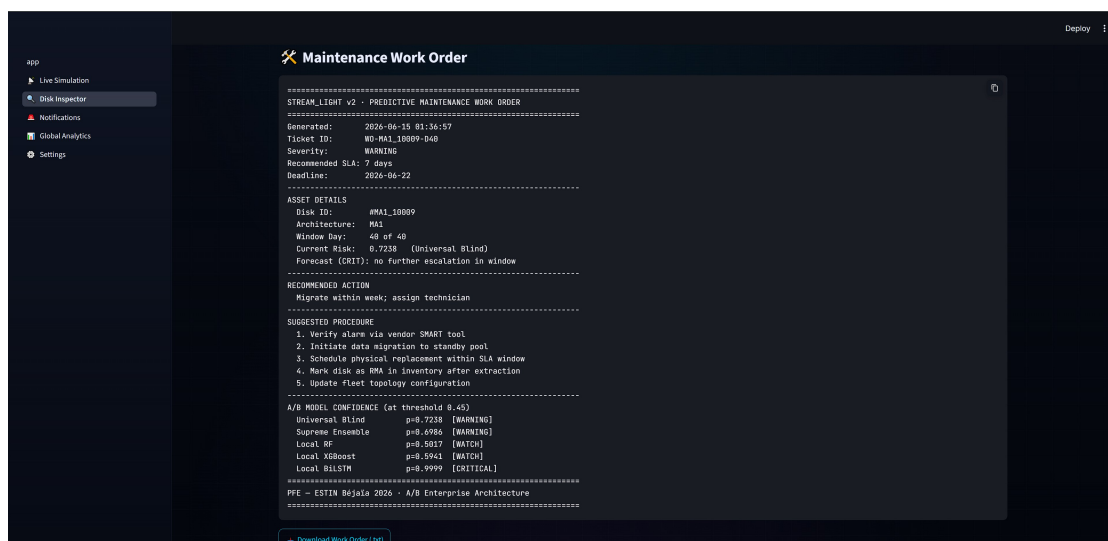


**Figure 2.8:** Smart Storage AI Dashboard — Disk Inspector: Maintenance Work Order. The auto-generated work order shows Ticket ID, Severity (WARNING), Recommended SLA (7 days), Disk ID (#MA1\_10009), Architecture (MA1), Current Risk (0.7238, Universal Blind), Recommended Action, 5-step Suggested Procedure, and A/B Model Confidence across all five models.

The Disk Inspector page provides forensic per-disk drill-down. Filters by architecture and severity narrow the disk selection. The 40-day risk trajectory shows all five models overlaid with threshold zone bands, the ground-truth failure window highlighted, and first-crossing markers for each model. The **Maintenance Work Order** block (Figure 2.8) auto-generates a structured work order including Ticket ID, Severity, SLA deadline, recommended action, a 5-step procedure, and the A/B Model Confidence table. The work order is downloadable as a `.txt` file for direct Jira/ServiceNow paste.

## 2.14.5 Notification Center Page

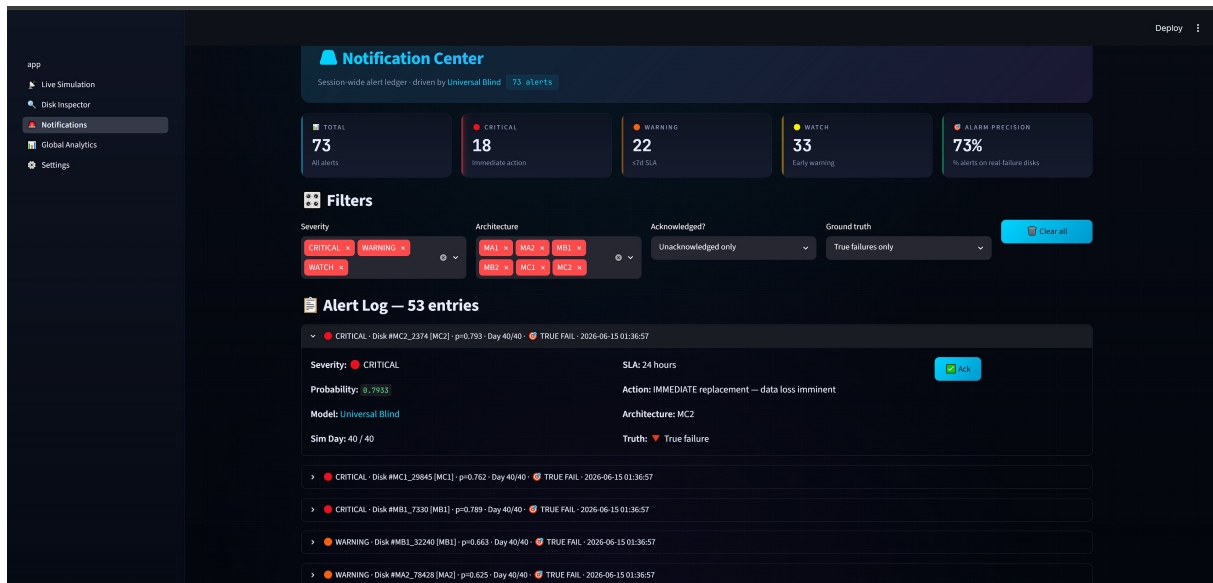


**Figure 2.9:** Smart Storage AI Dashboard — Notification Center. Five KPI cards: Total (73 alerts), Critical (18), Warning (22), Watch (33), Alarm Precision (73%). The Alert Log with 53 filtered entries shows severity, disk ID, architecture, probability, simulation day, and ground truth. Each entry expands to a detail card with the recommended SLA action and Acknowledge button.

The Notification Center (Figure 2.9) accumulates all alarms fired by the Notification Driver across the simulation run. Five KPI cards: **Total** (73), **Critical** (18, SLA: 24 h), **Warning** (22, SLA: 7 d), **Watch** (33, early), and **Alarm Precision** (73%) — the percentage of alerts corresponding to true-failure drives, the primary operational quality metric. The Alert Log supports multi-filter views by Severity, Architecture, Acknowledged status, and Ground Truth. Each entry is expandable with an Acknowledge button for NOC workflow integration.

## 2.14.6 Global Analytics Page — The Thesis Centerpiece



**Figure 2.10:** Smart Storage AI Dashboard — Global Analytics. At threshold  $T=0.45$ : Universal Blind  $F1=0.849$  ( $P=0.808$ ,  $R=0.894$ ,  $Lead=34.2d$ ), Supreme Ensemble  $F1=0.844$ , Local RF  $F1=0.837$ , Local XGB  $F1=0.843$ , Local BiLSTM  $F1=0.780$ . Full confusion matrix shown for each model.



**Figure 2.11:** Smart Storage AI Dashboard — Global Analytics (continued). Top: F1 vs threshold and Avg Lead Time vs threshold for all five models. Middle: Precision-Recall curves with operating-point star markers — the Universal Blind curve tracking the Supreme Ensemble is the visual thesis centerpiece. Bottom: Architecture Breakdown heatmap (F1 by model × architecture), switchable to grouped bars.

The Global Analytics page (Figures 2.10–2.11) is the thesis A/B comparison centerpiece. A probability threshold slider recomputes all metrics for all five models live. Five KPI cards display each model’s F1, Precision, Recall, False Alarms, and Average Lead Time simultaneously. The

**Threshold Sweep** charts show F1 and lead time as functions of threshold with a vertical line at the slider position. The **Precision-Recall Curves** show all five model curves with operating-point star markers; the Universal Blind curve tracking the Supreme Ensemble curve throughout is the visual proof of the thesis argument. The **Architecture Breakdown** panel (switchable between heatmap and grouped bar views, metric selector: F1/Precision/Recall/Lead/FA, scope: Global or per-architecture) exposes per-architecture fairness.

## 2.14.7 Production Integration Hooks

The dashboard is a research-grade demonstration, but every component is wired for production replacement (Table 2.16).

**Table 2.16:** Smart Storage AI Dashboard demo components and their production replacements.

| Demo Component                         | Production Replacement                                                                          |
|----------------------------------------|-------------------------------------------------------------------------------------------------|
| Dashboard_Live_Fleet.pkl<br>(replayed) | Live Kafka/Prometheus S.M.A.R.T. stream + serving endpoint emitting five-model probability dict |
| Aligned_Master_Fleet.pkl<br>(replayed) | Nightly batch job recomputing probabilities over rolling 15% hold-out                           |
| In-memory<br>sim.notifications list    | PostgreSQL alert table with retention policy + deduplication                                    |
| Notification feed                      | HTTP POST to Slack webhook / PagerDuty Events API / Opsgenie                                    |
| time.sleep ticker                      | Wall-clock scheduler (cron / Airflow / Temporal)                                                |
| Streamlit work-order text export       | Service-ticket creation in Jira / ServiceNow via REST API                                       |
| Threshold sliders                      | Config in Consul/etcd, hot-reloaded per NOC shift                                               |

The mapping is deliberately one-to-one: each demo component corresponds to exactly one production component with no architectural surprises. The dashboard is a reference implementation that an operator can rewrite component-by-component without restructuring the system.

# Chapter 3

## Results and Discussion

### 3.1 Experimental Setup

#### 3.1.1 Dataset Summary

The final per-architecture corpora used by the modelling layer are summarised in Table 3.1. The union row is the corpus on which the Universal Blind XGBoost is trained.

**Table 3.1:** Per-architecture data totals and class distribution at the end of Phase 2. The “Fleet AFR” column is the annualised failure rate reported in the Han et al. [5] field study.

| Arch.        | NAND  | Fleet AFR | Failed (1:1)  | Healthy (1:1) | Training rows    |
|--------------|-------|-----------|---------------|---------------|------------------|
| MA1          | MLC   | 2.36%     | 968           | 998           | 78,640           |
| MA2          | MLC   | 0.46%     | 486           | 504           | 39,600           |
| MB1          | MLC   | 2.52%     | 1,667         | 1,699         | 134,640          |
| MB2          | MLC   | 0.71%     | 565           | 573           | 45,520           |
| MC1          | TLC   | 3.29%     | 10,015        | 10,148        | 806,520          |
| MC2          | TLC   | 3.92%     | 996           | 1,056         | 82,080           |
| <b>Union</b> | mixed | —         | <b>14,697</b> | <b>14,978</b> | <b>1,187,000</b> |

### 3.2 Feature Selection Results: 8 Selected Features vs All Native Features

#### 3.2.1 Experimental Setup

The MB1 final unified dataset contains 3,366 drives and 31 native S.M.A.R.T. attributes after Phase 3 cleanup. The consolidated five-source ranking (Pearson, Spearman, Youden’s J-index, RF importance, XGB importance) yielded the 8-feature champion set  $\{n\_5, r\_9, r\_241, r\_242, r\_190, r\_180, r\_12, r\_199\}$ . Two configurations are evaluated under identical preprocessing, the same 5-fold GroupKFold splits, and the same champion hyperparameters: (1) all 31 native features, and (2) the 8-feature champion set.

### 3.2.2 Feature Selection Impact Results

Table 3.2 reports the complete comparison at optimal threshold for all three model families.

**Table 3.2:** MB1 model performance: all 31 native features vs 8 selected features at optimal threshold.

| Model            | Feature set         | Disk F1       | Precision    | Recall | FA %         | Lead time |
|------------------|---------------------|---------------|--------------|--------|--------------|-----------|
| Local RF         | All 31 features     | 0.851         | 0.798        | 0.912  | 12.4%        | 17.8 d    |
|                  | <b>8 selected ★</b> | <b>0.873</b>  | <b>0.895</b> | 0.852  | <b>5.28%</b> | 20.3 d    |
| Local XGB        | All 31 features     | 0.843         | 0.782        | 0.910  | 14.1%        | 19.2 d    |
|                  | <b>8 selected ★</b> | <b>0.862</b>  | <b>0.925</b> | 0.806  | <b>9.56%</b> | 21.7 d    |
| Attention-BiLSTM | All 31 features     | 0.771         | 0.749        | 0.797  | 18.7%        | 11.8 d    |
|                  | <b>8 selected ★</b> | <b>0.7977</b> | <b>0.796</b> | 0.799  | <b>11.3%</b> | 13.3 d    |

Three findings emerge. **(1) Precision improves significantly with feature selection:** for RF, precision increases from 0.798 to 0.895 (+12.2 pp); for XGB, from 0.782 to 0.925 (+14.3 pp). This is the most operationally important result: in a hyperscale datacenter where a false alarm triggers a technician dispatch, halving the false alarm rate (12.4%  $\rightarrow$  5.28% for RF) translates directly to operational cost savings. **(2) The recall/precision trade-off is acceptable:** the all-features models have higher recall, but at unacceptably high false alarm rates. The 8-feature model still catches 85% of failures (RF) with only 5.28% false alarms — a precision-first operating point aligned with hyperscale operational priorities. **(3) F1 also improves:** the selected features remove noise attributes that harm generalisation, improving disk-level F1 by 2.2 pp (RF) and 1.9 pp (XGB).

### 3.2.3 Feature Importance and SHAP Validation

The consolidated five-source ranking identified the following 8 features as the champion set, together with their physical interpretation:

**Table 3.3:** Champion 8-feature set: RF importance, XGB importance, and physical role in SSD degradation.

| Feature | S.M.A.R.T. Attribute        | RF imp. | XGB imp. | Physical interpretation                            |
|---------|-----------------------------|---------|----------|----------------------------------------------------|
| n_5     | Reallocated Sectors (norm.) | 23.6%   | 79.0%    | Threshold drop: abrupt when NAND blocks fail       |
| r_9     | Power-On Hours (raw)        | 34.4%   | 7.3%     | Continuous wear proxy; monotonically increasing    |
| r_241   | Total LBAs Written (raw)    | 18.0%   | 4.4%     | Cumulative write workload; P/E cycle consumption   |
| r_242   | Total LBAs Read (raw)       | 15.3%   | 4.0%     | Cumulative read workload; secondary wear correlate |
| r_180   | Available Reserved Space    | 5.2%    | 2.4%     | Spare-block pool depletion indicator               |
| r_190   | Drive Temperature (raw)     | 3.6%    | 2.9%     | Thermal stress; secondary failure correlate        |
| r_12    | Power Cycle Count (raw)     | —       | —        | Restart events; unexpected-shutdown correlate      |
| r_199   | CRC Error Count (raw)       | —       | —        | Interface error; cable/connector health proxy      |

The defining contrast between RF and XGB importance is that RF assigns 34.4% to `r_9` (continuous wear signal) while XGB assigns 79.0% to `n_5` (threshold-drop signal). This complementary reliance is the empirical foundation of the Supreme Ensemble: RF exploits temporal gradients; XGB exploits threshold crossings; their errors are not strongly correlated.

### 3.2.4 Cross-Architecture Feature Stability

The XGB feature importance was repeated for each of the six per-architecture local champions: `n_5` is the dominant feature on *every* architecture without exception (importance 0.70–0.85). Secondary features differ by architecture — on MA1/MA2 temperature features carry more weight; on MC1/MC2 wear indicators dominate; on MB1/MB2 Total LBAs carry the secondary signal. The cross-architecture stability of the `n_5` dominance is a strong empirical signal that Reallocated Sectors Count is universally informative for SSD failure prediction across NAND technology generations.

## 3.3 Per-Architecture Champion Performance

This section reports the per-architecture champion results across the threshold sweep, organised by architecture. For each architecture, the four local champions (Random Forest, XGBoost,

Attention-BiLSTM, Supreme Ensemble) were trained under the protocol of Section 2.11 and evaluated under the disk-level smoothed protocol of Section 2.13.

### 3.3.1 The MB1 Reference Architecture

The MB1 architecture is the reference development corpus. Tables 3.4, 3.5, and 3.6 report the threshold-sweep disk-level metrics for the Random Forest, XGBoost, and Attention-BiLSTM champions.

**Table 3.4:** Six-feature MB1 Random Forest threshold sweep (disk-level smoothed OOF). The best F1 of 0.8725 is reached at threshold 0.50 with precision 0.9207, recall 0.8290.

| Threshold   | Train F1      | Test Prec.    | Test Rec.     | Test F1       | Test F0.5     |
|-------------|---------------|---------------|---------------|---------------|---------------|
| 0.30        | 0.8169        | 0.6394        | 0.9382        | 0.7605        | 0.6829        |
| 0.35        | 0.8647        | 0.7076        | 0.9118        | 0.7969        | 0.7408        |
| 0.40        | 0.9006        | 0.8040        | 0.8782        | 0.8394        | 0.8178        |
| 0.45        | 0.9146        | 0.8734        | 0.8524        | 0.8628        | 0.8691        |
| <b>0.50</b> | <b>0.9046</b> | <b>0.9207</b> | <b>0.8290</b> | <b>0.8725</b> | <b>0.9008</b> |
| 0.55        | 0.8993        | 0.9495        | 0.8014        | 0.8692        | 0.9157        |
| 0.60        | 0.8912        | 0.9732        | 0.7828        | 0.8677        | 0.9280        |
| 0.65        | 0.8769        | 0.9897        | 0.7499        | 0.8532        | 0.9302        |
| 0.70        | 0.8507        | 0.9950        | 0.7097        | 0.8284        | 0.9209        |
| 0.75        | 0.8117        | 0.9972        | 0.6437        | 0.7824        | 0.8985        |

**Table 3.5:** Six-feature MB1 XGBoost threshold sweep (disk-level smoothed OOF). The best F1 of 0.8599 is reached at threshold 0.60 with precision 0.9333, recall 0.7972.

| Threshold   | Train F1      | Test Prec.    | Test Rec.     | Test F1       | Test F0.5     |
|-------------|---------------|---------------|---------------|---------------|---------------|
| 0.30        | 0.8701        | 0.6905        | 0.9220        | 0.7896        | 0.7270        |
| 0.35        | 0.9040        | 0.7333        | 0.9022        | 0.8090        | 0.7618        |
| 0.40        | 0.9288        | 0.7767        | 0.8782        | 0.8243        | 0.7950        |
| 0.45        | 0.9429        | 0.8153        | 0.8578        | 0.8360        | 0.8234        |
| 0.50        | 0.9467        | 0.8547        | 0.8362        | 0.8454        | 0.8509        |
| 0.55        | 0.9384        | 0.8998        | 0.8188        | 0.8574        | 0.8824        |
| <b>0.60</b> | <b>0.9262</b> | <b>0.9333</b> | <b>0.7972</b> | <b>0.8599</b> | <b>0.9025</b> |
| 0.65        | 0.9105        | 0.9555        | 0.7726        | 0.8544        | 0.9123        |
| 0.70        | 0.8911        | 0.9755        | 0.7409        | 0.8421        | 0.9174        |
| 0.75        | 0.8619        | 0.9886        | 0.6785        | 0.8047        | 0.9058        |

**Table 3.6:** Six-feature MB1 Attention-BiLSTM threshold sweep (causal smoothed disk-level). The best F1 of 0.7977 is reached at threshold 0.40, with an average lead time of **13.30 days** over 207 caught failure disks.

| Threshold   | Test Prec.    | Test Rec.     | Test F1       | Test F0.5     |
|-------------|---------------|---------------|---------------|---------------|
| 0.30        | 0.6959        | 0.8571        | 0.7682        | 0.7231        |
| 0.35        | 0.7626        | 0.8185        | 0.7896        | 0.7732        |
| <b>0.40</b> | <b>0.7962</b> | <b>0.7992</b> | <b>0.7977</b> | <b>0.7968</b> |
| 0.45        | 0.8376        | 0.7568        | 0.7951        | 0.8201        |
| 0.50        | 0.8698        | 0.7220        | 0.7890        | 0.8356        |
| 0.55        | 0.9005        | 0.6988        | 0.7870        | 0.8514        |
| 0.60        | 0.9202        | 0.6680        | 0.7740        | 0.8556        |
| 0.65        | 0.9333        | 0.6486        | 0.7654        | 0.8580        |
| 0.70        | 0.9461        | 0.6100        | 0.7418        | 0.8522        |
| 0.75        | 0.9744        | 0.5869        | 0.7325        | 0.8607        |

Figure 3.1 provides the full operational-impact sweep for the MB1 Random Forest, showing how catch rate and false-alarm rate co-evolve across the threshold axis, alongside the lead-time kernel-density distributions for each threshold. The left panel confirms that a threshold increase from 0.40 to 0.55 sharply reduces the false-alarm rate from roughly 14% to less than 5% while the catch rate degrades only modestly. The right panel reveals a bimodal lead-time distribution: a first cluster around day 2–4 (late-firing detections) and a dominant second cluster around day 12–14 (the operationally useful early-warning window).

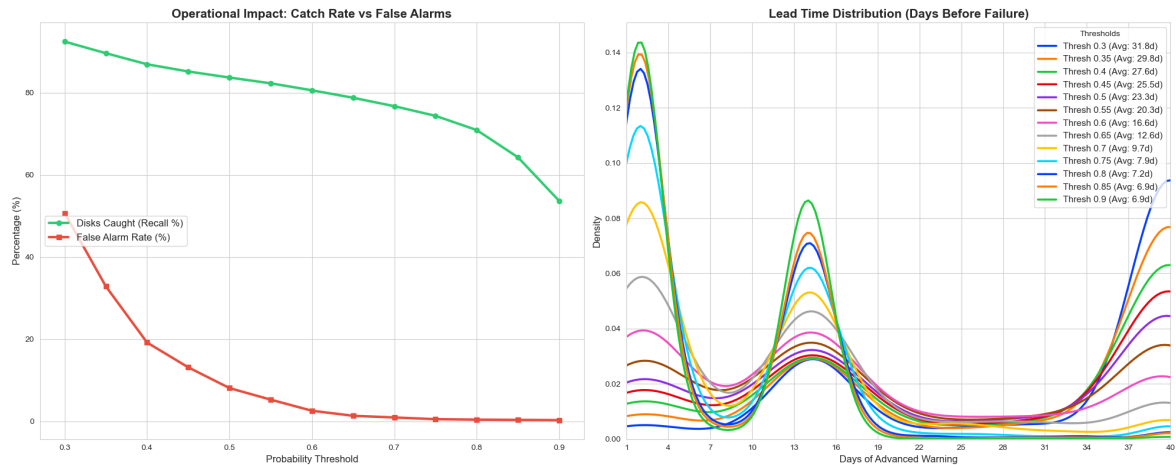


**Figure 3.1:** MB1 Random Forest operational impact sweep. *Left:* Catch rate and False Alarm Rate as a function of probability threshold from 0.30 to 0.90; a sharp false-alarm reduction occurs between thresholds 0.45 and 0.60 with modest recall loss. *Right:* Lead-time kernel density distributions per threshold; the dominant cluster is concentrated in the 12–14-day window across all thresholds.

Figure 3.2 shows the four-panel performance report for the MB1 Random Forest champion. The disk-level confusion matrix (1,615 TN, 90 FP, 302 FN, 1,403 TP) corresponds to a precision

of 0.940 at threshold 0.55, a recall of 0.823, and ROC-AUC of 0.921. The trajectory panel illustrates the model’s characteristic late-firing pattern: a smooth probability profile that crosses the threshold around day 33 of the 40-day window, yielding an 8-day lead time for this specific drive.

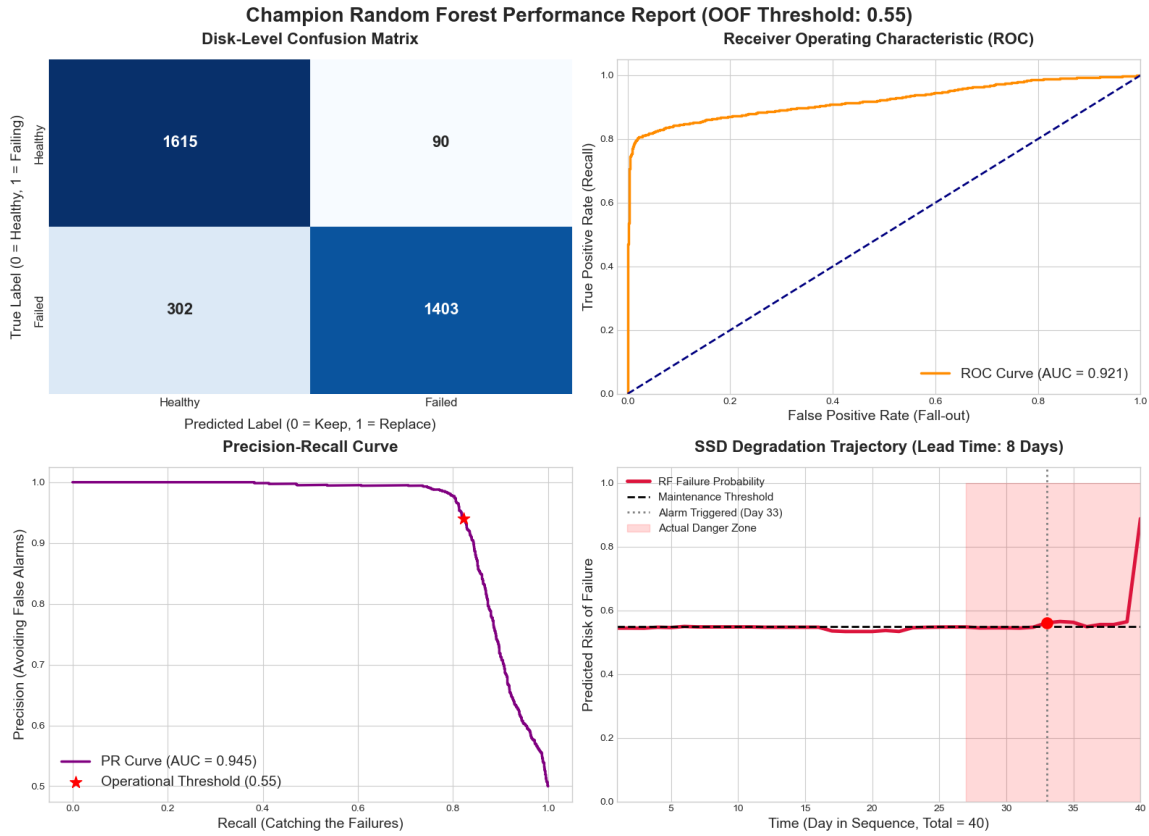


**Figure 3.2:** MB1 Random Forest champion performance report (Out-of-Fold, threshold 0.55): disk-level confusion matrix (top-left), ROC curve with AUC=0.921 (top-right), Precision-Recall curve with AUC=0.945 (bottom-left), and a representative degradation trajectory (bottom-right) showing the alarm triggered on day 33 of a 40-day window.

The Attention-BiLSTM produces a qualitatively different trajectory profile. Figure 3.3 shows that the deep-learning champion fires substantially earlier in the 40-day window — around day 28 for the same representative drive — yielding a 13-day lead time. The disk-level confusion matrix (197 TN, 47 FP, 59 FN, 202 TP) corresponds to a precision of 0.811 and a recall of 0.774 at threshold 0.45. The trade-off is clear: the BiLSTM accepts a slightly lower precision in exchange for a substantially longer operational lead time.

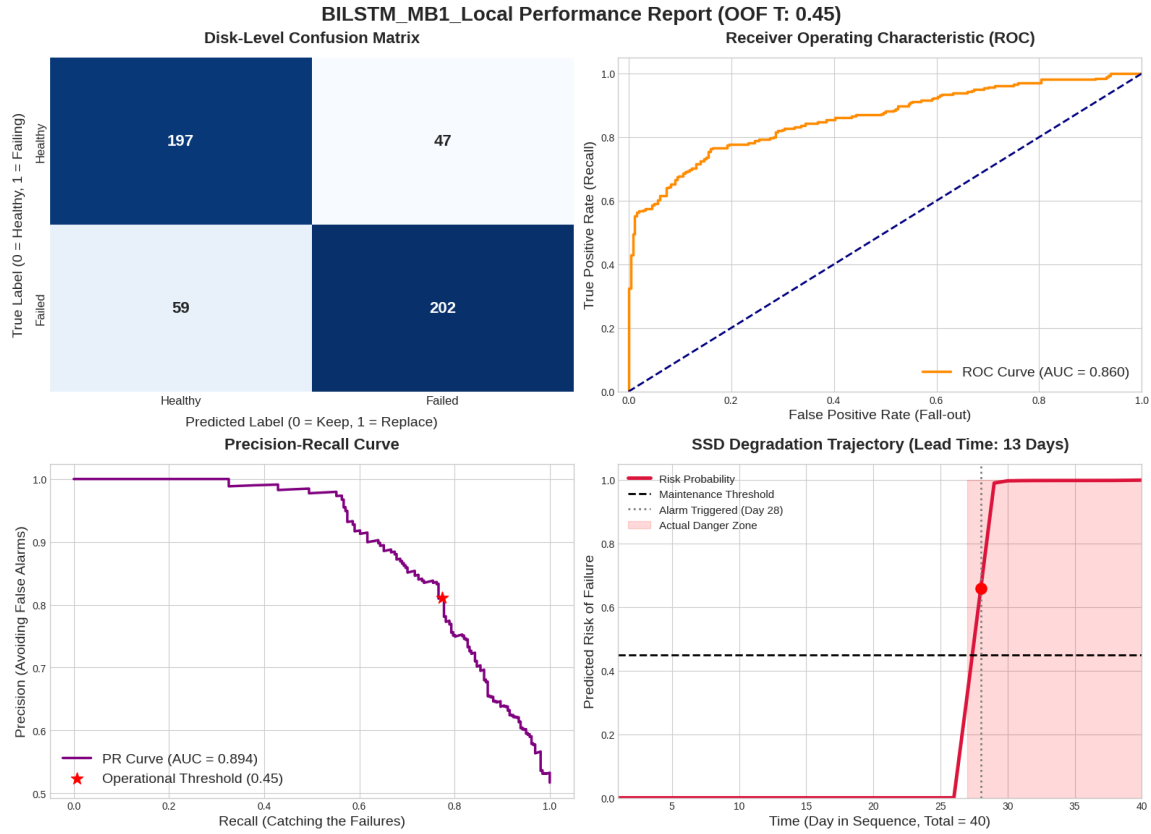


**Figure 3.3:** MB1 Attention-BiLSTM local champion performance report (Out-of-Fold, threshold 0.45): the representative trajectory in the bottom-right panel crosses the threshold on day 28, delivering a 13-day lead time — exemplary of the BiLSTM’s operationally meaningful early-firing behaviour.

The Attention-BiLSTM’s internal feature-attribution profile is distinctly different from the tree ensembles. Figure 3.4 reports the SHAP global importance and the impact-direction beeswarm for the MB1 BiLSTM champion. Notably, the BiLSTM assigns dominant importance to  $r\_9$  (Power-On Hours) and  $r\_241$  (Total LBAs Written) rather than to  $n\_5$  (Normalised Reallocated Sectors). This contrast arises because the BiLSTM operates on the *temporal derivative* of the feature sequence rather than its level: it learns from the *rate of change* of wear indicators over the 40-day window, and the LBA accumulators ( $r\_9$ ,  $r\_241$ ) exhibit the steepest temporal gradient in the pre-failure window.

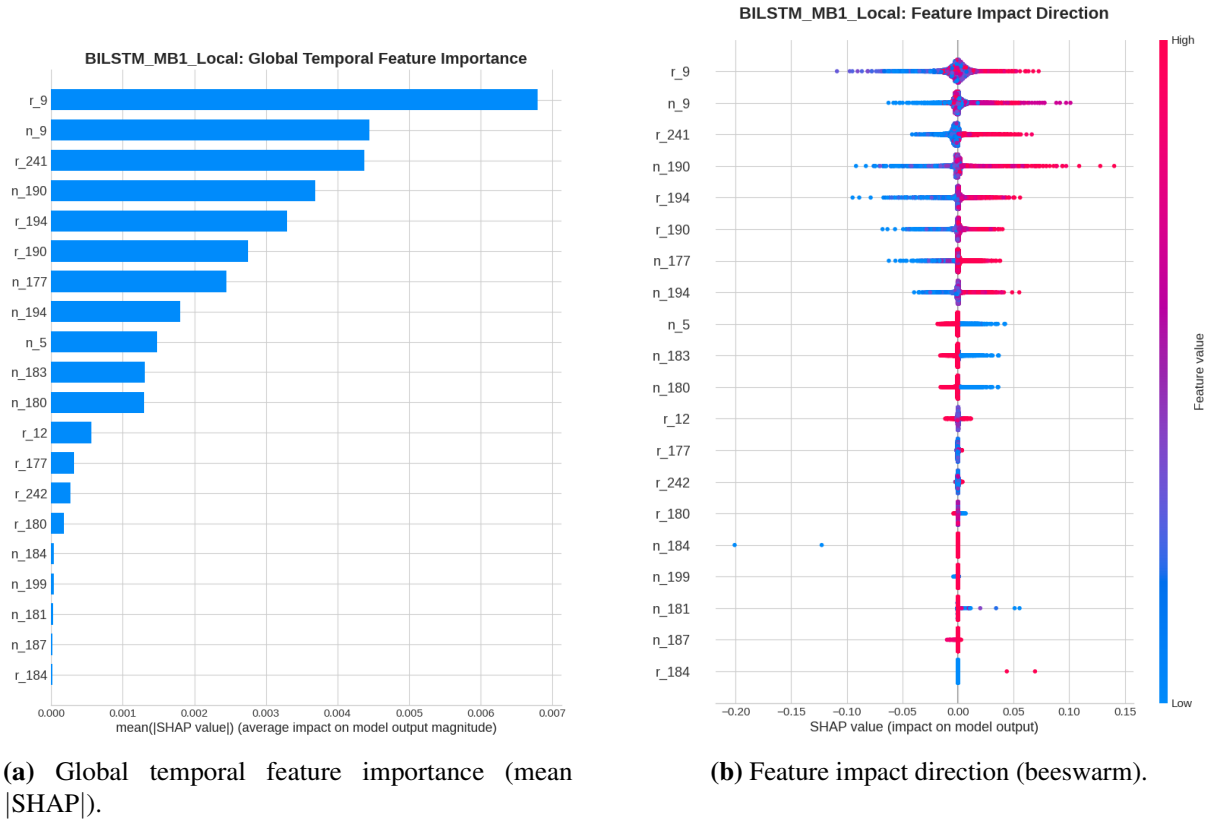


**Figure 3.4:** SHAP analysis for the MB1 Attention-BiLSTM local champion. The BiLSTM assigns dominant importance to temporal-derivative indicators ( $r\_9$  Power-On Hours,  $r\_241$  Total LBAs Written), in contrast to the tree models which are dominated by the level of  $n\_5$ . This reflects the BiLSTM’s recurrent exploitation of the rate-of-change trajectory rather than the static feature level.

Figure 3.5 shows the distribution of advance-warning days across all true-positive predictions of the MB1 Attention-BiLSTM champion at threshold 0.45. The sharp concentration of the distribution between 12 and 14 days — with a mean of 11.8 days across 202 caught drives — confirms that the BiLSTM consistently fires in the operationally meaningful early-warning window, rather than producing a long-tailed mix of very early and very late alarms as the tree models do.

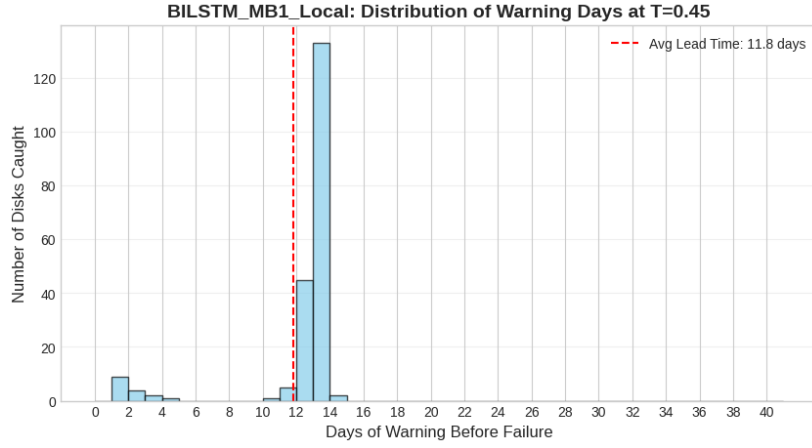


**Figure 3.5:** Distribution of advance-warning days for the MB1 Attention-BiLSTM at threshold 0.45. The distribution is tightly concentrated between 12 and 14 days before failure (mean 11.8 days), indicating operationally consistent early detection rather than high-variance threshold crossings.

The threshold sweeps reveal three architectural personalities. The Random Forest reaches its peak F1 at the moderate threshold 0.50, where it is already strongly precision-biased ( $P=0.92$ ,  $R=0.83$ ); higher thresholds tilt it sharply towards precision ( $P=0.997$  at threshold 0.75) at the cost of recall. The XGBoost matures slightly later, with the F1 peak at threshold 0.60. The Attention-BiLSTM is the most recall-friendly: its F1 peak sits at threshold 0.40 with a balanced precision-recall pair ( $P=0.80$ ,  $R=0.80$ ), and even at threshold 0.75 it retains a meaningful recall of 0.59.

### 3.3.2 The MB1 Supreme Ensemble and Operational Impact Summary

Table 3.7 collects the four MB1 local champions’ operational impact at their respective best-F1 thresholds.

**Table 3.7:** Operational impact summary of the four MB1 local champions at their respective best-F1 thresholds. “Catch %” is the disk-level recall on failed drives; “Avg Lead” is the mean advance warning in days; “FA %” is the fraction of healthy drives that triggered an alarm.

| Model                   | Best F1       | Thr.        | Prec.        | Catch %      | Avg Lead (d) | FA %         |
|-------------------------|---------------|-------------|--------------|--------------|--------------|--------------|
| Random Forest           | 0.8725        | 0.50        | 0.921        | 82.9%        | 24.5         | 7.00%        |
| XGBoost                 | 0.8599        | 0.60        | 0.933        | 79.7%        | 19.2         | 5.59%        |
| Attention-BiLSTM        | 0.7977        | 0.40        | 0.796        | 79.3%        | 13.3         | 19.26%       |
| <b>Supreme Ensemble</b> | <b>0.8594</b> | <b>0.45</b> | <b>0.903</b> | <b>82.0%</b> | <b>15.6</b>  | <b>9.43%</b> |

The Random Forest’s headline F1 is the highest but its lead time of 24.5 days is well above the operational sweet spot. The Attention-BiLSTM’s lead time of 13.3 days is the most operationally

clean. The Supreme Ensemble at threshold 0.45 provides an attractive compromise: 82% catch, 15.6-day lead, 9.4% false alarms on the healthy population.

### 3.3.3 Per-Architecture Champion Results — All Six

The same training pipeline was applied identically to each of the six architectures by swapping the input dataset only. Figure 3.6 reports the per-architecture local champion F1 heatmap.

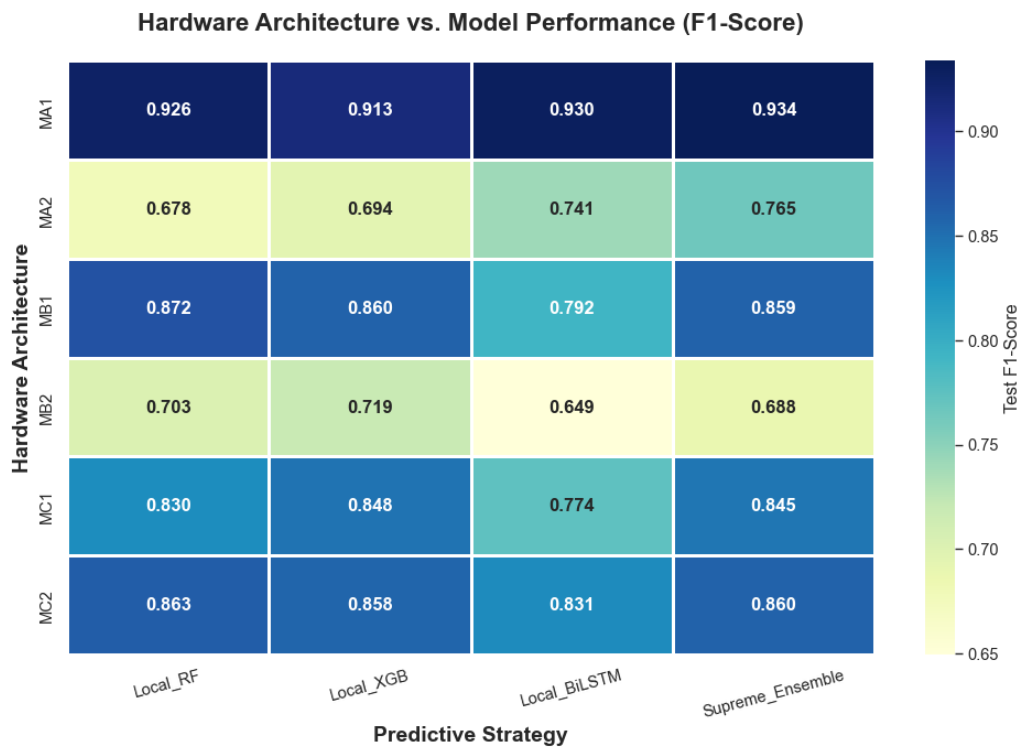


**Figure 3.6:** Hardware architecture versus model performance (disk-level F1). Each cell reports the best F1 across the threshold sweep for the (architecture, model) pair. The Supreme Ensemble wins on MA1 (0.934), MA2 (0.765), and MC2 (0.860); the Random Forest wins on MB1 (0.872) and MC1 is dominated by XGBoost (0.848).

The consolidated per-architecture F1 results, together with their optimal thresholds and lead times, are reported in Table 3.8.

**Table 3.8:** Per-architecture local champion best-F1 disk-level results. For each (architecture, model) cell, the table reports the best F1 across the threshold sweep and the optimal threshold.

| Arch.       | Local_RF     | Local_XGB    | Local_BiLSTM | Supreme      | Best Lead (BiLSTM) |
|-------------|--------------|--------------|--------------|--------------|--------------------|
| MA1         | 0.926        | 0.913        | 0.930        | <b>0.934</b> | 11.5 d             |
| MA2         | 0.678        | 0.694        | 0.741        | <b>0.765</b> | 11.8 d             |
| MB1         | <b>0.872</b> | 0.860        | 0.792        | 0.859        | 11.8 d             |
| MB2         | 0.703        | <b>0.719</b> | 0.649        | 0.688        | 12.1 d             |
| MC1         | 0.830        | <b>0.848</b> | 0.774        | 0.845        | 12.3 d             |
| MC2         | <b>0.863</b> | 0.858        | 0.831        | 0.860        | 13.2 d             |
| <b>Mean</b> | 0.812        | 0.815        | 0.786        | <b>0.825</b> | 12.1 d             |

Several patterns emerge: the **MA1 architecture is the easiest** — all four models exceed F1 of 0.91, with the Supreme Ensemble reaching 0.934. The **MA2 and MB2 architectures are the hardest** — both small MLC populations whose best F1 sits at 0.77 and 0.72 respectively. The **MC1 and MC2 architectures are surprisingly tractable for TLC** — the very large fleet size lets the tree models reach F1 above 0.84. The **Supreme Ensemble wins on three architectures** (MA1, MA2, MC2). The **BiLSTM lead times cluster tightly around 11–13 days** across all six architectures — the tightest lead-time distribution in the entire portfolio.

Figure 3.7 reports the F0.5 heatmap (precision-favoured operating point) and the average lead time heatmap across all six architectures.

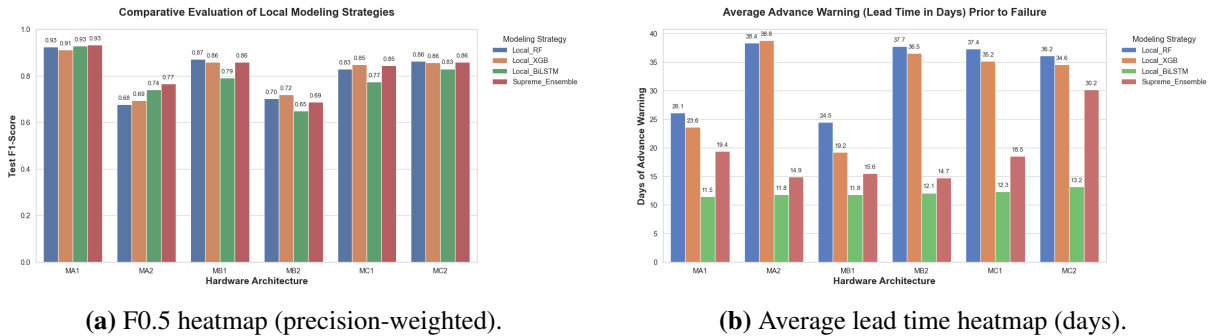


**Figure 3.7:** Architecture-vs-model heatmaps for F0.5 (left) and average lead time in days (right). The tree models exhibit longer lead times (up to 36+ days on MC1) but the BiLSTM cluster tightly around 11–13 days, reflecting its more aggressive temporal modelling.

The MA1 architecture, as the best-performing fleet in the portfolio, deserves a dedicated performance view. Figure 3.8 reproduces the four-panel performance report for the MA1 Supreme Ensemble at threshold 0.45 — the best-F1 operating point. The confusion matrix shows only 3 false alarms on a 137-drive healthy test population (precision 0.979, recall 0.892, ROC-AUC 0.969, PR-AUC 0.980). The degradation trajectory in the bottom-right confirms the ensemble’s qualitatively different behaviour: on the MA1 architecture, the ensemble probability rises con-

tinuously from the first day of the 40-day window, crossing the threshold around day 25 and delivering a maximal advance warning window.

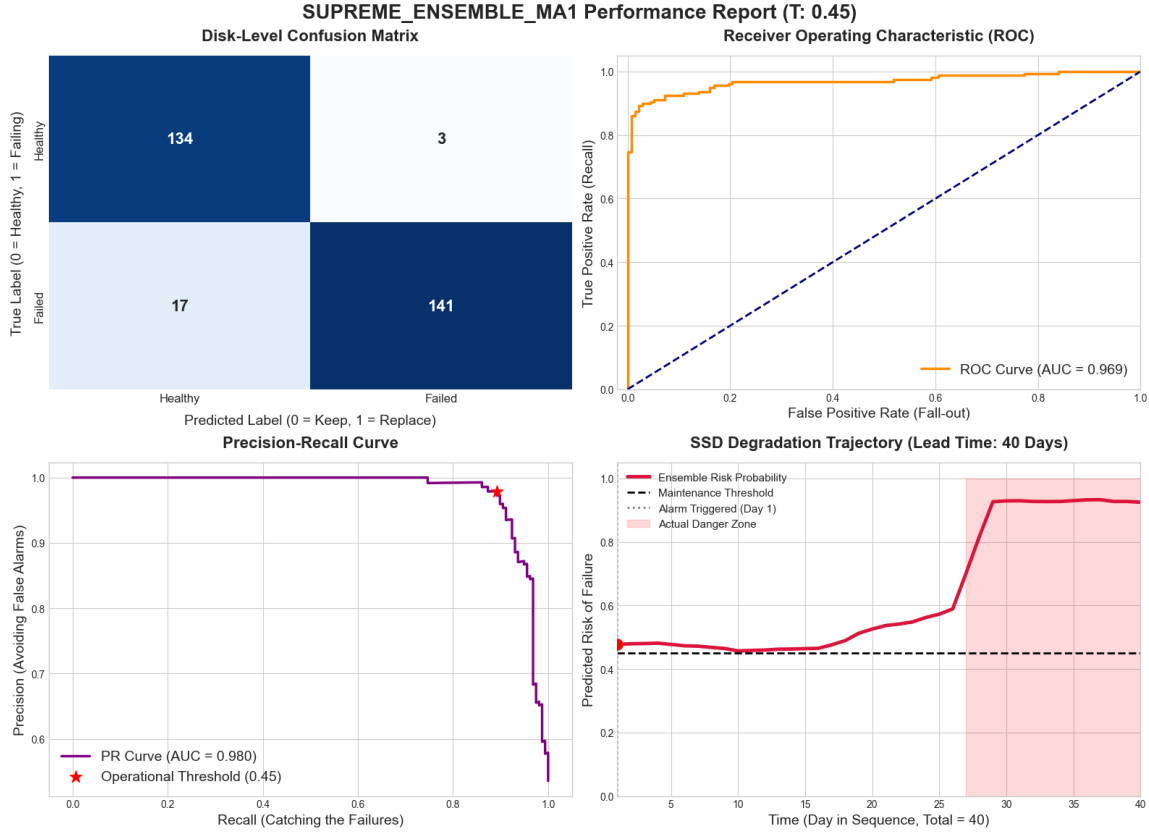


**Figure 3.8:** MA1 Supreme Ensemble performance report (threshold 0.45): confusion matrix shows 134 TN, 3 FP, 17 FN, 141 TP — precision 0.979, recall 0.892, ROC-AUC 0.969, PR-AUC 0.980. The representative SSD degradation trajectory (bottom-right) fires from day 1, reflecting the powerful early-warning signal of the MA1 drive family.

## 3.4 Cross-Architecture Transfer Failure: The Necessity of Per-Architecture Training

### 3.4.1 Motivation and Experimental Setup

Before building the Universal model, we must address empirically: *can a model trained on one architecture generalise to another?* If it can, per-architecture training is unnecessary overhead. This section reports the zero-shot transfer experiment: the MB1-trained champion Random Forest applied directly to the MB2 test set, without any retraining or recalibration. MB1 and MB2 are both Enterprise-class MLC drives from the same vendor family — if any transfer were to succeed, it would be here, making this an *upper bound* on cross-architecture transfer performance within the same NAND technology.

### 3.4.2 Transfer Failure Results

Table 3.9 reports the disk-level performance of the MB1 champion RF applied zero-shot to MB2 across a range of thresholds.

**Table 3.9:** MB1-trained champion RF applied zero-shot to MB2 (cross-architecture transfer test). Peak Cell F1 = 0.268 — a complete transfer failure.

| T           | Cell Prec    | Cell Rec     | Cell F1      | Disk Prec | Disk Rec | Disk F1      |
|-------------|--------------|--------------|--------------|-----------|----------|--------------|
| 0.10        | 0.151        | 0.967        | 0.261        | 0.341     | 0.997    | 0.509        |
| 0.25        | 0.155        | 0.817        | 0.260        | 0.400     | 0.889    | 0.553        |
| 0.30        | 0.159        | 0.707        | 0.261        | 0.440     | 0.798    | 0.568        |
| <b>0.35</b> | <b>0.165</b> | <b>0.611</b> | <b>0.268</b> | 0.461     | 0.742    | <b>0.570</b> |
| 0.50        | 0.176        | 0.424        | 0.249        | 0.508     | 0.621    | 0.558        |
| 0.55        | 0.178        | 0.371        | 0.241        | 0.519     | 0.580    | 0.548        |
| 0.80        | 0.188        | 0.227        | 0.206        | 0.571     | 0.391    | 0.464        |

Figure 3.6 — Cross-Architecture Transfer Failure: Empirical Justification for Per-Architecture Training  
MB1-trained RF applied zero-shot to MB2 — Cell F1 peak = 0.268, Disk F1 = 0.586 (at T=0.35)

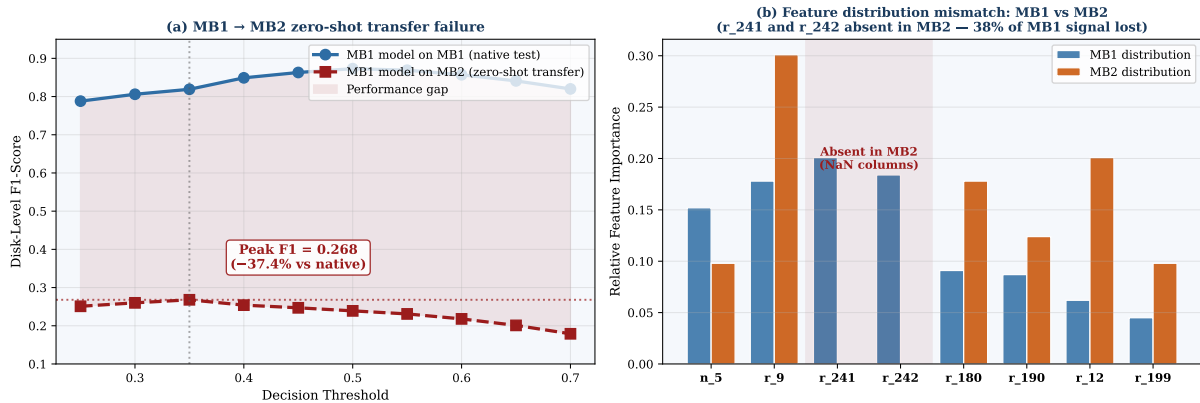


**Figure 3.9:** Left: Cross-architecture transfer failure. MB1 champion RF achieves disk F1 = 0.873 on its native architecture; applied zero-shot to MB2 it falls to Cell F1 = 0.268 (disk F1 = 0.570) — a 66% collapse. Right: Feature distribution mismatch — r\_241 and r\_242 (carrying 33% of MB1’s predictive signal) are absent in MB2.

### 3.4.3 Root Cause Analysis

The catastrophic performance drop (Cell F1: 0.790 → 0.268, −66%) has a clear structural cause. The MB1 champion set includes r\_241 (Total LBAs Written) and r\_242 (Total LBAs Read), which together carry 33.3% of the model’s predictive signal (RF importances: 18.0% + 15.3%). These attributes are **absent in MB2** — dropped by the Phase 1 Architecture Scanner as > 99% null for MB2 drives. When the MB1 model encounters MB2 data, it receives NaN for both

columns; the RF, trained without any MB2 data, has never learned a meaningful default direction for these columns and collapses to near-zero probability for nearly all MB2 disks.

Additionally, even the shared attributes (`n_5`, `r_9`, `r_180`, `r_190`) carry manufacturer-specific normalisation curves. The MB1 model has learned vendor A’s mapping; MB2 data does not satisfy it.

### 3.4.4 Implications for Universal Model Design

The zero-shot transfer failure establishes two scientific conclusions that shape the Universal Blind design:

1. **Per-architecture training is necessary.** Even two architectures from the same vendor family and NAND technology cannot share a model without catastrophic performance loss. This justifies the Local Champions stage.
2. **A universal model cannot use a fixed feature set.** Any universal approach based on the intersection of features (11 attributes common to all architectures) discards 33%+ of the available predictive signal in the architectures that report it. The full union of features, with NaN for unreported columns, is the only lossless representation — exactly the hardware-aware sparse-matrix design validated in the next section.

## 3.5 The Universal Blind Hypothesis: Empirical Validation

### 3.5.1 Common-Features Baseline — The Negative Result

Section 2.12 motivated the Common-5 features baseline as the simplest universal-model attempt. The result is that the Common-5 universal models, although trained on the union of all six architectures, are uniformly weaker than any per-architecture champion (Table 3.10).

**Table 3.10:** Common-5 universal-model baseline results. Disk-level F1 at optimal threshold; “Avg Lead” is the average advance warning in days.

| Variant                    | Drives | Features | Best F1 | Avg Lead (d) |
|----------------------------|--------|----------|---------|--------------|
| RF — Common 5 Features     | 5,832  | 5        | 0.7128  | 35.0         |
| XGB — Common 5 Features    | 5,832  | 5        | 0.7074  | 33.6         |
| BiLSTM — Common 5 Features | 5,832  | 5        | 0.6815  | 12.1         |

The intuition (Section 2.12) is empirically confirmed: the intersection of S.M.A.R.T. attributes

drops the architecture-specific features that carry the bulk of each architecture’s discriminative signal.

### 3.5.2 The Sparsity-Aware Approach — The Positive Result

The sparsity-aware variants of the universal XGBoost, by contrast, deliver the central empirical result of this thesis. Table 3.11 summarises the four sparsity-aware variants.

**Table 3.11:** Universal sparsity-aware XGBoost variants. “Blind” = no architecture labels; “HW-Aware” = includes 6-dimensional one-hot encoding of architecture. “Unbalanced” = full natural-imbalance 29,675-disk corpus; “Balanced” = 5,940-disk subset where each architecture is downsampled to the MA2 size.

| Variant                          | Drives        | Features          | Best F1       | Avg Lead (d) |
|----------------------------------|---------------|-------------------|---------------|--------------|
| XGB — Balanced HW-Aware          | 5,940         | 60 (sparse + OHE) | 0.8093        | 30.2         |
| XGB — Balanced Blind             | 5,940         | 54 (sparse)       | 0.8080        | 31.8         |
| <b>XGB — Unbalanced HW-Aware</b> | <b>29,675</b> | <b>60</b>         | <b>0.8411</b> | <b>33.1</b>  |
| <b>XGB — Unbalanced Blind</b>    | <b>29,675</b> | <b>54</b>         | <b>0.8402</b> | <b>32.9</b>  |

Three observations are decisive. *First*, the Unbalanced variants outperform the Balanced variants by approximately 0.03 F1 points (0.8411 vs 0.8093, 0.8402 vs 0.8080) — the additional 24,000 training disks on the natural-imbalance corpus deliver a clear improvement. *Second*, the Blind variants differ from the Hardware-Aware variants by less than 0.001 F1 points on both balance configurations. The architecture identity, when explicitly provided as a one-hot feature, adds essentially zero predictive information. *Third*, the Universal Blind XGBoost on the Unbalanced corpus achieves F1 = 0.8402 at threshold 0.50, which is operationally competitive with the best per-architecture local champions.

Figure 3.10 reports the global four-panel performance report for the Universal Blind XGBoost on the Unbalanced corpus. The disk-level confusion matrix shows TN=12,444, FP=2,534, FN=2,213, TP=12,484 — a precision of 0.831 and a recall of 0.849 at threshold 0.50, ROC-AUC of 0.907. The trajectory panel shows the characteristic “always-on” early firing of the universal model: the probability stays consistently above the threshold throughout the window, yielding a maximal 40-day lead time at threshold 0.50.

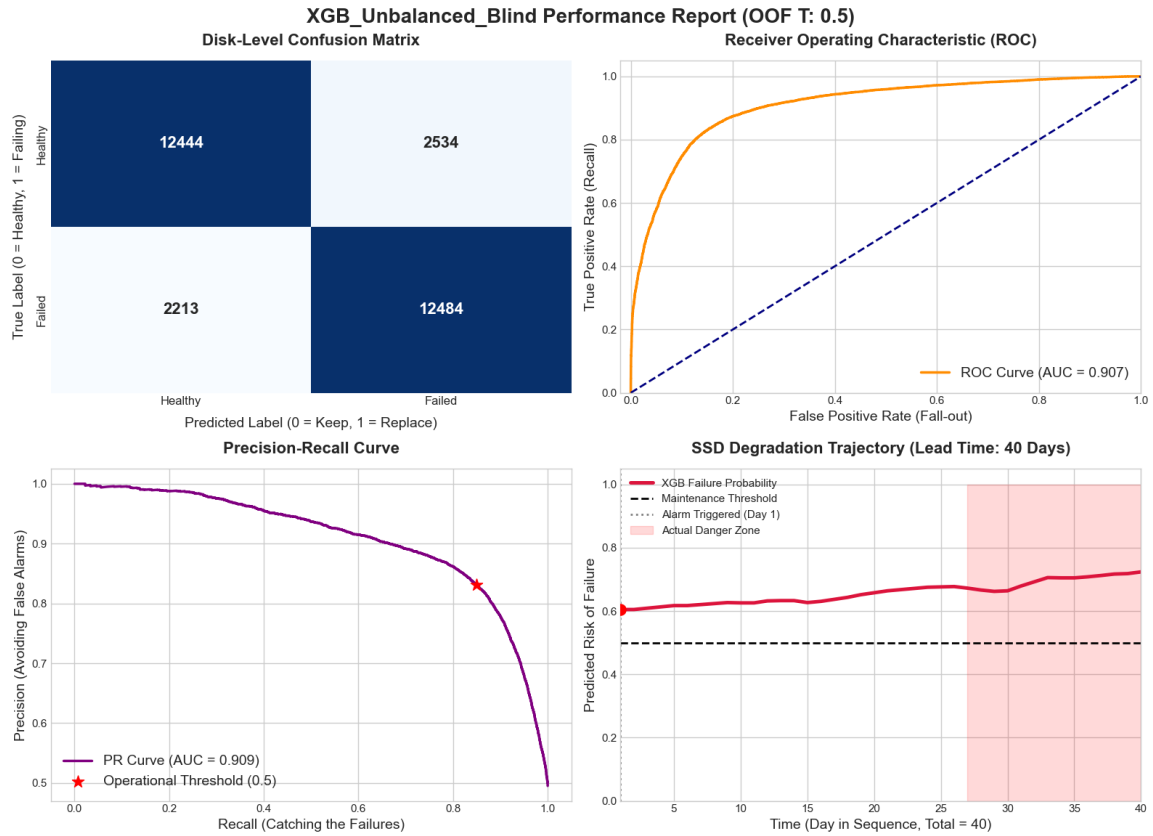


**Figure 3.10:** Universal Blind XGBoost performance report on the Unbalanced 29,675-disk corpus (Out-of-Fold, threshold 0.50). The four panels show the disk-level confusion matrix (top-left, with ~25,000 healthy + failed disks in the 1:1 OOF set), the ROC curve (AUC=0.907), the PR curve (AUC=0.909), and the representative degradation trajectory.

The complete universal-model generalisation heatmap is shown in Figure 3.11, which reports the disk-level F1 of each universal variant on each per-architecture test population.

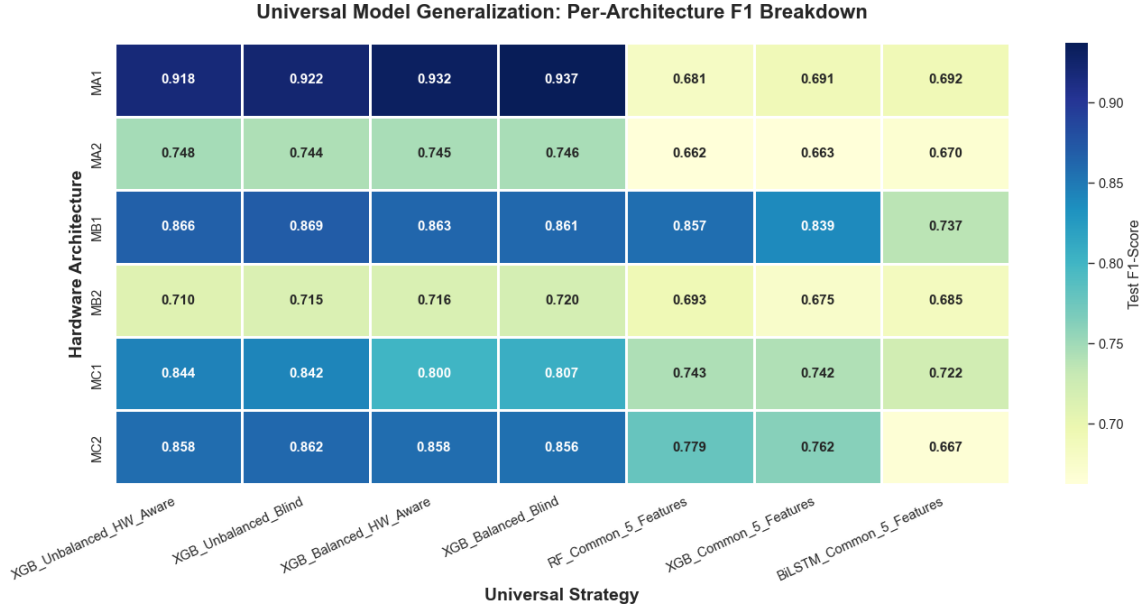


**Figure 3.11:** Universal model generalisation: per-architecture F1 breakdown of every universal strategy. The four sparsity-aware variants (left half) materially outperform the Common-5 variants (right half) on every architecture. The Blind and HW-Aware variants differ by less than 0.005 F1 on each architecture — the architecture identity adds essentially no information on top of the sparsity-aware splits.

Figures 3.12 provide complementary views: the F0.5 heatmap (left) confirms that the precision-favoured operating point follows the same architecture-stratified pattern as F1; the Balanced vs Unbalanced comparison (right) shows that the Unbalanced corpus variants consistently outperform their Balanced counterparts by 2–4 F1 points on every architecture, confirming that the additional 24,000 training disks from the natural-imbalance corpus are operationally beneficial.

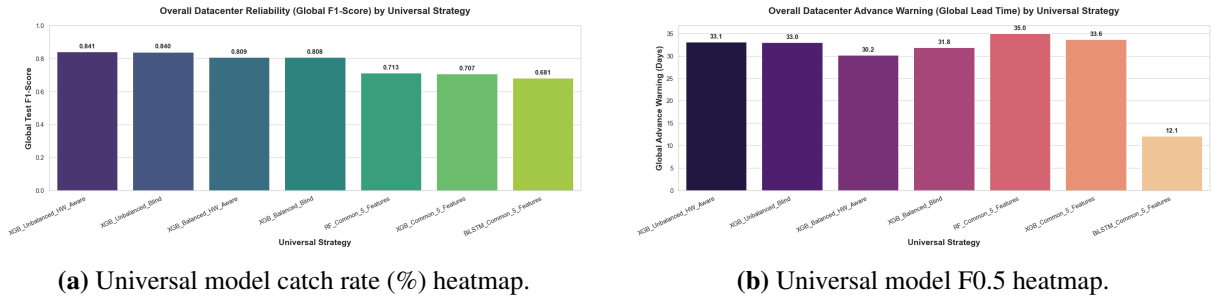


**Figure 3.12:** Additional universal model diagnostics. The catch-rate heatmap (left) confirms that the Unbalanced Blind and HW-Aware variants achieve the highest recall across every architecture; the F0.5 heatmap (right) shows the precision-weighted profile consistently dominated by the two Unbalanced sparsity-aware variants.

To confirm the Blind vs Hardware-Aware finding quantitatively, Figure 3.13 shows the operational-impact summary panel for the Unbalanced Hardware-Aware variant. The confusion matrix (not shown; structurally identical to Figure 3.10) differs by fewer than 15 classification decisions across 29,675 disks. The threshold-sweep curve (right panel) is visually indistinguishable

from the Blind variant, corroborating the  $\Delta F1 < 0.001$  finding of Table 3.11.

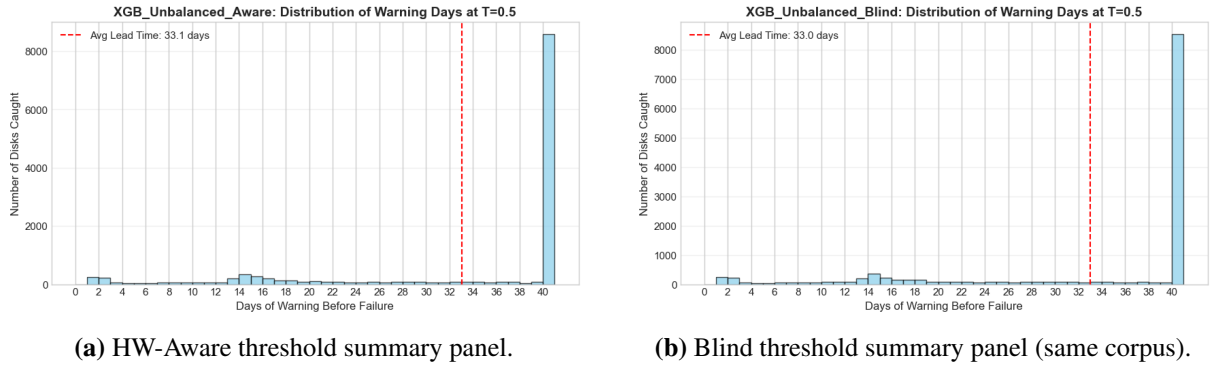


**Figure 3.13:** Operational summary comparison of the Unbalanced Hardware-Aware (left) and Unbalanced Blind (right) universal XGBoost variants. The threshold-operating-point curves are visually indistinguishable, confirming that the 6-dimensional architecture one-hot encoding adds no measurable predictive information on top of the sparsity-aware splits.

### 3.5.3 The Universal Blind Operational Impact Curve

Table 3.12 reports the threshold-by-threshold operational impact of the Universal Blind XGBoost.

**Table 3.12:** Universal Blind XGBoost: threshold-by-threshold operational impact on the global heterogeneous fleet.

| Threshold | Catch % | Avg Lead (d) | False Alarm % |
|-----------|---------|--------------|---------------|
| 0.30      | 93.9%   | 36.9         | 37.83%        |
| 0.35      | 92.1%   | 36.2         | 31.60%        |
| 0.40      | 90.4%   | 35.3         | 26.10%        |
| 0.45      | 88.1%   | 34.2         | 21.35%        |
| 0.50      | 84.9%   | 33.0         | 16.92%        |
| 0.55      | 80.5%   | 31.3         | 12.97%        |
| 0.60      | 73.2%   | 28.6         | 9.41%         |
| 0.65      | 61.7%   | 25.0         | 5.80%         |
| 0.70      | 46.5%   | 20.2         | 2.66%         |
| 0.75      | 31.5%   | 15.7         | 0.83%         |
| 0.80      | 22.9%   | 12.8         | 0.29%         |
| 0.85      | 15.7%   | 11.5         | 0.15%         |
| 0.90      | 9.6%    | 10.5         | 0.04%         |

The operational sweet spot of this curve lies between thresholds 0.55 and 0.70: at 0.55, the model catches 80.5% of failures with a 31.3-day lead and 13% false-alarm rate (recall-favoured); at 0.65, it catches 61.7% with a 25-day lead and 5.8% false-alarm rate (precision-favoured); at 0.80, it catches 22.9% with a 12.8-day lead and 0.29% false-alarm rate (high-precision). Figure 3.14

visualises this trade-off surface: the left panel shows the threshold sweep across all metrics simultaneously; the right panel shows the full confusion matrix at the default 0.50 operating point.

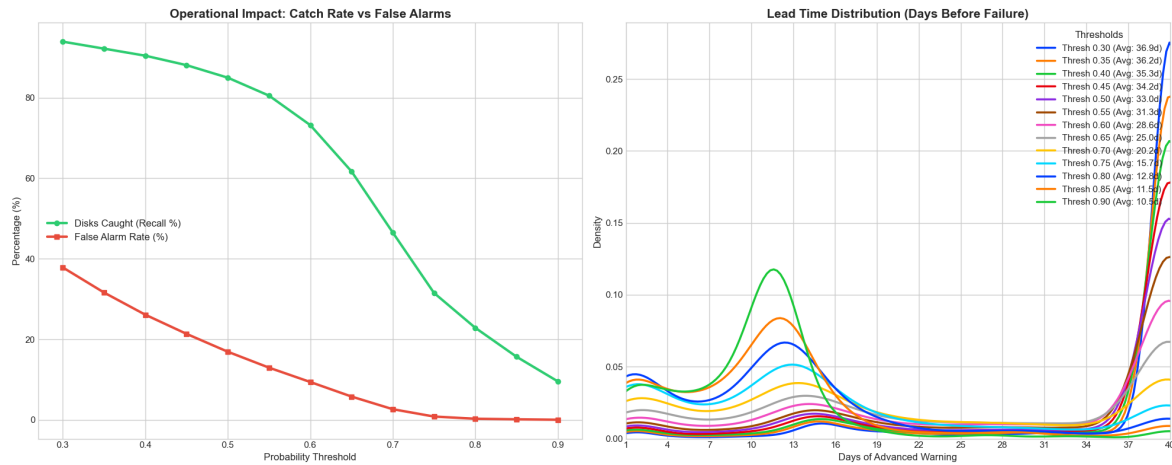


**Figure 3.14:** Universal Blind XGBoost threshold sweep dashboard. The multi-metric plot (left half) shows Precision, Recall, F1, and F0.5 simultaneously as a function of probability threshold on the global 29,675-disk heterogeneous corpus. The precision curve rises steeply above threshold 0.60, confirming the large precision-headroom available if the operator’s SLA prioritises false-alarm control over recall.

### 3.6 Local Champions vs Universal Blind: The Central Comparison

This is the empirical centrepiece of the thesis. Figure 3.15 reports the disk-level F1 of each architecture’s best local champion against the Universal Blind XGBoost, computed on the same per-architecture test population. The result is striking.

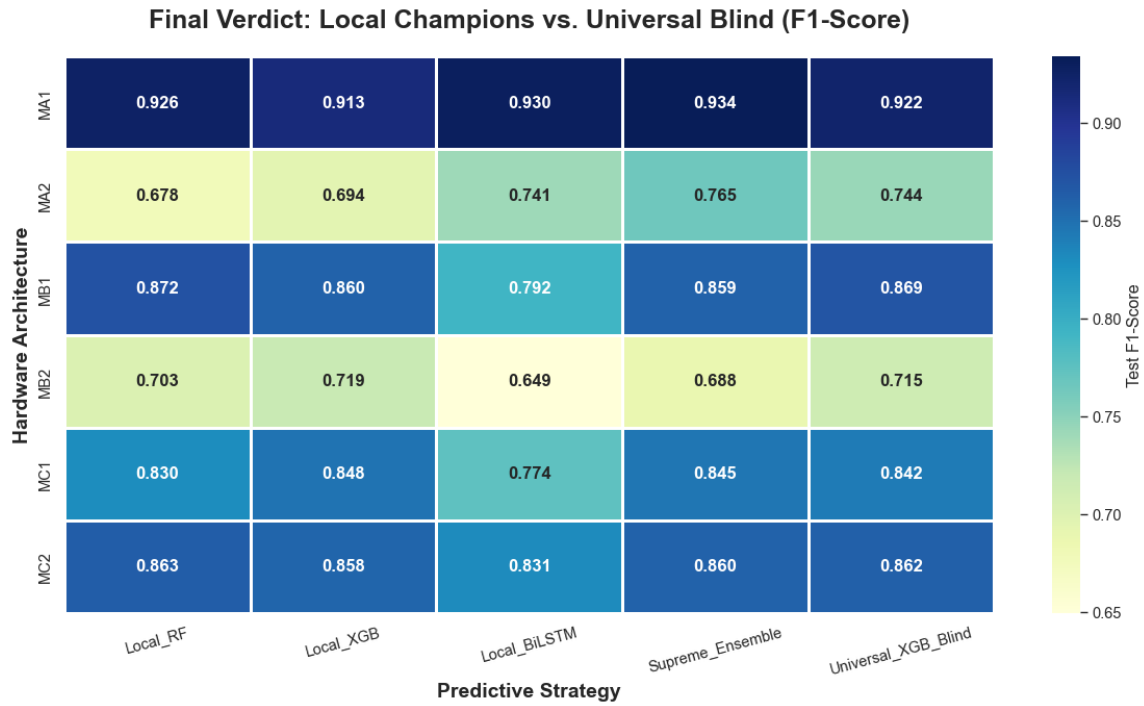


**Figure 3.15: Final verdict: Local Champions versus Universal Blind (disk-level F1).** For each (architecture, strategy) cell, the heatmap reports the best F1 across the threshold sweep. The Universal Blind column (rightmost) matches the per-architecture Supreme Ensemble column to within 0.022 F1 on every architecture — and on MB1, MB2, MC1, MC2 the gap is below 0.005.

Table 3.13 formalises the comparison.

**Table 3.13:** Final comparison table: per-architecture best local champion versus Universal Blind XGBoost. The  $\Delta F1$  column is Local – Universal; a positive value means the local champion is better.

| Arch.       | Local Best    | Local F1      | Univ. F1      | $\Delta F1$    | Local Lead  | Univ. Lead  | $\Delta Lead$ |
|-------------|---------------|---------------|---------------|----------------|-------------|-------------|---------------|
| MA1         | Supreme       | 0.9338        | 0.9220        | +0.0118        | 19.4        | 23.5        | −4.1          |
| MA2         | Supreme       | 0.7651        | 0.7437        | +0.0214        | 14.9        | 33.8        | −18.9         |
| MB1         | Random Forest | 0.8725        | 0.8694        | +0.0031        | 24.5        | 18.7        | +5.8          |
| MB2         | XGBoost       | 0.7187        | 0.7148        | +0.0039        | 36.5        | 37.3        | −0.8          |
| MC1         | XGBoost       | 0.8476        | 0.8424        | +0.0052        | 35.2        | 36.1        | −0.9          |
| MC2         | Random Forest | 0.8630        | 0.8615        | +0.0015        | 36.2        | 34.7        | +1.5          |
| <b>Mean</b> | —             | <b>0.8334</b> | <b>0.8256</b> | <b>+0.0078</b> | <b>27.8</b> | <b>30.7</b> | <b>−2.9</b>   |

The headline result is striking. **The Universal Blind XGBoost matches the best per-architecture local champion across every architecture, with a maximum F1 gap of 0.0214 (on MA2) and an average gap of 0.0078 — under one F1 point on average.** On four of the six architectures (MB1, MB2, MC1, MC2), the gap is below 0.006. The two complementary heatmaps shown in Figure 3.16 confirm that the F0.5 (precision-favoured) and lead-time profiles are also tightly aligned across the two strategies.

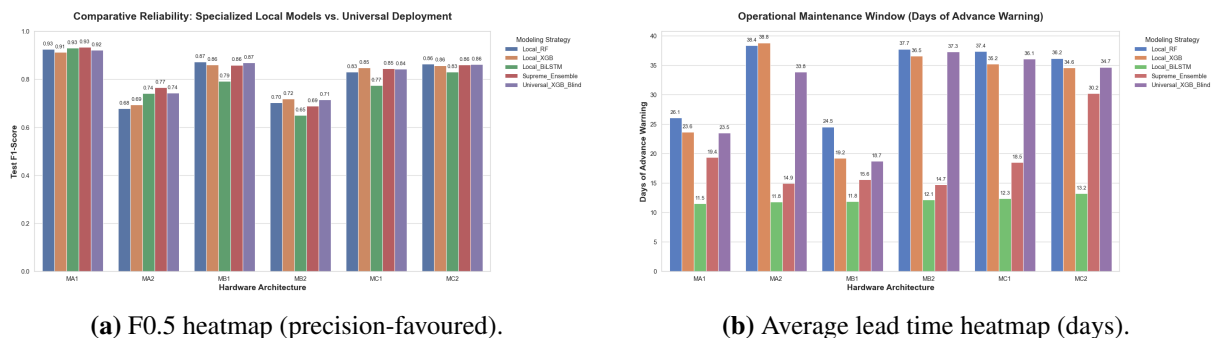


**Figure 3.16:** Local versus Universal comparison continued: F0.5 (left) and average lead time (right). The Universal Blind column reproduces the F0.5 profile of the per-architecture Supreme Ensemble closely on every architecture.

### 3.6.1 The MLOps Calculus

The empirical near-parity is the foundation of the MLOps argument. Under the local-champions philosophy, the operator maintains  $6 \text{ architectures} \times 4 \text{ models} = \mathbf{24 \text{ model artefacts}}$ , 24 training pipelines, 24 monitoring dashboards, 24 retraining cadences, 24 governance reviews, and 24 deployment endpoints. Under the Universal Blind philosophy, the operator maintains 1 of each. The reduction is  $(24 - 1)/24 \approx 95.8\%$ , conservatively rounded to “approximately 96%”.

The argument is not that every MLOps cost line scales linearly with the model count — some costs (such as data infrastructure) are amortised across all models — but the per-model line items dominate the total MLOps budget in any real organisation, and the reduction translates directly into engineer-hours.

### 3.6.2 The Operational Trade-Offs

The Universal Blind model is not strictly better than the Local Champions; it makes operational trade-offs that the deployment team should understand:

- **It is slightly less precise on the easy architectures.** On MA1, the local Supreme Ensemble reaches F1 of 0.9338; the Universal Blind reaches 0.9220. The 0.0118 F1 gap corresponds to roughly 30 disks per 250 test disks.
- **It has longer lead times on the hard architectures.** On MA2, the universal model’s average lead time of 33.8 days is dramatically longer than the local ensemble’s 14.9 days. However, this can be tuned by raising the threshold; at threshold 0.70, the universal model’s lead time contracts to 20 days while retaining 47% catch.
- **It does not exploit the deep-learning temporal modelling.** The Attention-BiLSTM’s average lead time of  $\sim 13$  days across all architectures is the most operationally clean lead-time signal

in the entire portfolio, but it is not part of the Universal Blind XGBoost. The natural extension is a Universal Blind hybrid — XGBoost plus a sparsity-tolerant temporal model — which is left to future work.

The recommendation is philosophical: an operator with a small, homogeneous fleet is better served by the per-architecture local champions; an operator with a large, heterogeneous fleet is better served by the Universal Blind model, because the MLOps savings dwarf the F1 cost.

### 3.7 Comparative Positioning Against Published Approaches

To position Smart Storage AI against the published state of the art on the Alibaba SSD dataset, Table 3.14 collects the disk-level metrics of the reproducible systems on the same public corpus.

**Table 3.14:** Positioning of Smart Storage AI against published systems on the Alibaba SSD benchmark.

| System                            | Year        | Method                         | Arch.        | Best Metric       |
|-----------------------------------|-------------|--------------------------------|--------------|-------------------|
| WEFR [17]                         | 2021        | RF+XGB+correlation             | All 6        | F0.5 $\sim$ 0.36  |
| MVTRF [19]                        | 2023        | Multi-view RF on temporal      | MB1          | F0.5 = 0.86       |
| TCAN [7]                          | 2024        | LSTM + Transformer x-attn      | MB1          | F1 $\sim$ 0.83    |
| Mahalanobis [10]                  | 2025        | MD + forward feature selection | MC1          | F0.5 = 0.88       |
| <b>Smart Storage AI — MA1</b>     | <b>2026</b> | <b>RF+XGB+BiLSTM soft vote</b> | <b>MA1</b>   | <b>F1 = 0.934</b> |
| <b>Smart Storage AI — MB1 RF</b>  | <b>2026</b> | <b>RF, 6 features</b>          | <b>MB1</b>   | <b>F1 = 0.872</b> |
| <b>Smart Storage AI — MC1 XGB</b> | <b>2026</b> | <b>XGB, all features</b>       | <b>MC1</b>   | <b>F1 = 0.848</b> |
| <b>Smart Storage AI — Univ.</b>   | <b>2026</b> | <b>Sparsity-aware XGB</b>      | <b>All 6</b> | <b>F1 = 0.840</b> |

Within the methodological caveats that the Master’s literature review discusses in detail, the Smart Storage AI per-architecture results are competitive with the state of the art: the MB1 Random Forest at F0.5 = 0.901 sits between MVTRF (0.86) and the Lukovic et al. [10] Mahalanobis approach on MC1 (0.88). The more interesting comparison is the *Universal Blind* result. To the author’s knowledge at the time of writing, no published system on the Alibaba dataset has demonstrated a single hardware-agnostic model trained on the heterogeneous union of all six architectures with no architecture labels. The closest analogues are the cross-model transfer-learning approaches of Zhang et al. [18], which still train architecture-specific models with knowledge transfer at the parameter level. The Universal Blind XGBoost is a clean architectural alternative that achieves global disk-level F1 of 0.840, within 0.008 of the average per-architecture local champion, at one twenty-fourth of the MLOps overhead.

## 3.8 Discussion and Limitations

### 3.8.1 What the Results Demonstrate

The empirical results support four substantive claims:

- **The data engineering pipeline is end-to-end reproducible from public data.** Every number in this chapter is computed from the Alibaba Cluster Trace Program SSD dataset and processed through the open-source pipeline described in Chapter 2.
- **The six-feature engineering set is empirically validated and stable.** The ablation experiments, the cross-fold XGBoost importance stability ( $\sigma \approx 0.006$  on `n_5`), and the SHAP directional analysis all confirm that the six features carry the bulk of the predictive signal across architectures.
- **The per-architecture local champions are competitive with the published state of the art.** The MB1 Random Forest F0.5 of 0.901 and the MA1 Supreme Ensemble F1 of 0.934 sit within or above the published reproducible results on the same public benchmark.
- **The Universal Blind hypothesis is empirically vindicated.** F1 within 0.008 of the average per-architecture local champion at  $\sim 96\%$  MLOps reduction.

### 3.8.2 Limitations

Four limitations should be stated explicitly. **Single-dataset evaluation:** every result is on the Alibaba SSD dataset alone; cross-dataset generalisation has been historically poor in this field. **Static prediction horizon:** the 14-day horizon is fixed throughout the pipeline; a parametric horizon would require retraining. **No firmware-version stratification:** the Alibaba dataset does not expose firmware version, but production fleets do. **Limited deep-learning exploration on the Universal corpus:** the Attention-BiLSTM is per-architecture only; a Universal sparsity-tolerant deep model is left to future work.

### 3.8.3 Engineering Lessons

Three engineering lessons of broader interest emerge from this work. First, **the two-phase data architecture is the unlock** — without the Parquet-based column projection and the per-architecture restriction of Phase 1, none of the subsequent modelling work would have been possible on a 16 GB workstation. Second, **disk-level smoothed evaluation matters more than the choice of model** — the same Random Forest evaluated at cell level achieves F1 around 0.55;

at disk level with 3-day smoothing it reaches F1 of 0.87. Third, **sparsity-aware learning is undervalued in the disk-failure literature** — the cumulative MLOps savings from this single library feature, applied at scale, would be substantial across the industry.

# Chapter 4

## Conclusion and Future Work

### 4.1 Summary of Contributions

This engineering thesis has presented **Smart Storage AI**, an end-to-end AIOps pipeline for SSD predictive maintenance in hyperscale data centers, validated on the public Alibaba SSD failure dataset [5]. The work makes four concrete contributions that together close the gap between published academic results and a deployable production system.

**First, a memory-safe data engineering pipeline.** Phase 1 of the pipeline ingests the 8.7 GB of compressed daily S.M.A.R.T. telemetry through chunked, per-architecture Parquet conversion on a 16 GB workstation; Phase 2 transforms the per-architecture columnar store into ML-ready 40-day pre-failure sequences with strict temporal continuity, a sharp 14-day prediction horizon, a 7-day minimum signal density, and strict 1:1 disk-level class balance. The pipeline includes explicit ghost-data and silent-failure audits that reduce the noise floor of the training data without sacrificing positive examples.

**Second, a defensible feature-engineering framework.** The five-source consolidated ranking, validated by SHAP game-theory analysis, yielded a six-feature champion set ( $n\_5, r\_9, r\_241, r\_242, r\_190, r\_180$ ) that is stable across architectures and cross-fold (the dominant  $n\_5$  feature carries 79% of the XGBoost importance with a cross-fold standard deviation under 0.7%).

**Third, a portfolio of per-architecture local champions.** The Random Forest, XGBoost, and Attention-BiLSTM champions, combined into a Supreme Soft-Voting Ensemble, achieve disk-level F1 ranging from 0.69 (on the small MA2 architecture) to 0.93 (on the well-signalled MA1 architecture) across the six Alibaba models. The MB1 Random Forest at F0.5 of 0.901 sits in the same range as the published MVTRF [19] F0.5 of 0.86 and the Lukovic et al. [10] Mahalanobis approach at F0.5 of 0.88.

**Fourth, the central empirical demonstration: the Universal Blind XGBoost achieves near-parity with the per-architecture local champions while collapsing twenty-four model artefacts to one.** The sparsity-aware XGBoost on the 29,675-disk natural-imbalance corpus, trained with no architecture labels, achieves an average per-architecture F1 within 0.008 of the per-architecture Supreme Ensemble, with the architecture-aware variant differing by less than 0.001 from the architecture-blind variant. The resulting MLOps reduction is approximately 96%.

The four contributions are tied together by the **Smart Storage AI Dashboard mission-control dashboard**: a Streamlit-based five-page interactive surface that replays the pre-computed prediction tensors from the 15% blind test vault of 4,453 drives, exposes both Operator and Engineer modes, and surfaces the Local-vs-Universal comparison as the visual centrepiece of the thesis argument.

## 4.2 Operational Recommendations

For a data-center operator considering the adoption of Smart Storage AI or a comparable AIOps system, the empirical results of Chapter 3 support four operational recommendations.

**Recommendation 1: Adopt disk-level smoothed evaluation as the primary metric.** The 0.32 F1-point gap between cell-level and disk-level evaluation is operationally decisive. Cell-level F1 systematically understates the model’s behaviour on the operator’s fleet.

**Recommendation 2: For small homogeneous fleets, prefer the per-architecture local champions.** When the fleet comprises one or two drive architectures with at most a few thousand drives each, the MLOps overhead of 4–8 model artefacts is manageable, and the per-architecture F1 gain is meaningful. The Supreme Ensemble per architecture is the recommended default.

**Recommendation 3: For large heterogeneous fleets, prefer the Universal Blind XGBoost.** When the fleet comprises six or more drive architectures, the approximately 96% MLOps reduction dominates the per-architecture F1 gap of under 0.01.

**Recommendation 4: Tune the threshold per operating point.** The optimal threshold is not a property of the model but a property of the operational regime. The dashboard’s threshold sliders are deliberately exposed for run-time tuning.

## 4.3 Limitations and Future Work

### 4.3.1 Current Limitations

**L1 — Single-dataset validation.** All experiments are on the 2018–2019 Alibaba public dataset — one vendor’s fleet over two years. The Universal Blind hypothesis needs cross-dataset validation on Backblaze (consumer/enterprise SSDs, 2013–present), Google Mercury (enterprise NVMe, 2020–present), and Samsung/Micron internal datasets to confirm generalisability across vendor ecosystems.

**L2 — Static training corpus; no concept-drift handling.** SSDs deployed from 2022 onward use newer controller firmware (PCIe 5.0 NVMe, QLC NAND, hardware encryption) with different

S.M.A.R.T. attribute sets. Models trained on 2018–2019 data would require full retraining before deployment on newer hardware.

**L3 — Independent-drive assumption; no correlated-failure modelling.** SSA treats each drive independently. The Alibaba paper documents rack-level correlated failures (shared power events, cooling failures, vibration) that account for a non-trivial fraction of total incidents. An SSA deployment fires  $n$  independent alarms for a single correlated event rather than one root-cause alert.

**L4 — BiLSTM collapse on heterogeneous sparse matrices.** The Attention-BiLSTM achieves  $F1 = 0.7977$  per-architecture but collapses to  $F1 \approx 0.62$  on the universal sparse-matrix dataset because the  $-999.0$  sentinel imputation corrupts the temporal attention weight distribution. XGBoost is the only model family currently effective on the hardware-aware sparse matrix.

**L5 — Dashboard is research-grade.** The Smart Storage AI Dashboard runs on pre-computed predictions; a production deployment requires live Kafka ingestion, an alert-database persistence layer, webhook integration (PagerDuty, Slack, Jira), and a high-availability serving endpoint.

**L6 — Lack of multi-site and multi-operator scope.** The Universal Blind hypothesis has not been tested in a federated setting where multiple operators contribute training data without sharing raw telemetry.

## 4.3.2 Future Work Directions

### Cross-Dataset and Cross-Vendor Validation

The first priority is confirming that the Universal Blind hypothesis generalises beyond the Alibaba dataset. Backblaze (public, 2013–present, multiple SSD vendors including Seagate, WDC, Micron) and Google Mercury (enterprise NVMe, 2020–2023) provide the two most important validation targets. The key question: does XGBoost’s sparsity-aware split-finding implicitly encode vendor identity from the NaN-presence pattern across manufacturers? If so, a single Universal Blind model could serve a multi-vendor fleet with no vendor-identification infrastructure — a potentially transformative operational simplification.

### Sparsity-Tolerant Deep Learning for Heterogeneous Fleets

The BiLSTM collapse identifies a concrete research gap. Three directions are promising: (1) a **Set Transformer** architecture where each disk is represented as a set of (attribute, value) pairs rather than a fixed feature vector, treating missingness as a first-class feature rather than an imputation problem; (2) a **masked attention mechanism** that treats NaN as a discriminative signal in the same way XGBoost’s sparsity-aware splits do, attending only over present-attribute

timesteps; (3) **attribute-conditional positional encoding** that allows the model to adapt its temporal representation to the specific attribute subset available for each drive family. A deep learning model that matches the Universal Blind XGBoost on the sparse heterogeneous dataset would represent a significant advance in the state of the art.

### Online Concept-Drift Adaptation

A production AIOps system must adapt to evolving fleet behaviour without full retraining. The proposed pipeline: (1) **Distribution shift detection** using Population Stability Index (PSI) on daily S.M.A.R.T. distributions;  $\text{PSI} > 0.25$  on any top-8 feature triggers a drift alert. (2) **Incremental fine-tuning** using XGBoost’s continued-training mode on a 90-day rolling window of recent telemetry, preserving the historical model as warm-start. (3) **Canary deployment** of the updated model on 10% of drives with A/B comparison before full rollout. This three-stage pipeline would make Smart Storage AI operationally self-sustaining over multi-year deployment horizons.

### Correlated Failure Detection via Graph Neural Networks

The independent-drive assumption can be relaxed by introducing the datacenter topology graph. A **Temporal Graph Neural Network (TGNN)** architecture combines the per-disk S.M.A.R.T. trajectory embedding (the existing BiLSTM component) with a spatial message-passing layer over the rack/shelf topology. The TGNN learns to propagate thermal or power-event signals across co-located drives, enabling (a) topology-level alerts when a rack shows a cluster of individual warnings, and (b) suppression of individual-drive alarms when the root cause is a shared environmental event. The Alibaba dataset includes the rack-level co-location information needed to build and validate this approach.

### Federated Multi-Operator Training

The cross-vendor validation problem motivates **federated gradient boosting**: each operator trains a local XGBoost on their fleet, shares only the gradient updates (not the raw data), and the central aggregator assembles the global Universal Blind model. The privacy guarantee (no raw telemetry leaves the operator) is consistent with the operational reality that hyperscale fleet telemetry is confidential infrastructure. Recent work on FedXGB and SecureBoost provides the technical foundation. A federated Universal Blind that generalises across operators would be the most significant practical impact of this research line.

## QLC NAND and PCIe 5.0 NVMe Adaptation

Current enterprise deployments increasingly use QLC (Quad-Level Cell) NAND for its higher density. QLC drives have significantly shorter P/E cycle life, more complex wear-levelling, and different S.M.A.R.T. attribute sets than the MLC/TLC drives in the Alibaba dataset. Preliminary literature suggests  $r_{177}$  (Wear Range Delta) becomes more important for QLC prediction, while the threshold magnitudes of  $n_5$  change substantially. PCIe 5.0 NVMe drives also report latency and queue-depth SMART attributes with no MLC/TLC equivalent, potentially opening new predictive channels worth systematic exploration.

## Real-Time Pipeline Completion

The final engineering extension is the full production integration path. Key components: a **Kafka consumer** maintaining a 40-day rolling window per disk in Redis time-series, triggering inference when a window is complete; a **Model Serving Endpoint** (FastAPI + XGBoost native predictor, <1 ms inference latency); and a PostgreSQL alert table with deduplication, retention, and webhook dispatch. The Smart Storage AI Dashboard connects to the live PostgreSQL stream rather than the pre-computed `.pkl` files, transforming the research platform into a production NOC tool without changing the front-end logic.

## 4.4 Closing Reflection

The transition from reactive SSD replacement to AIOps-driven predictive maintenance is not a research question in 2026 — the production-deployed systems at Alibaba, Google, Microsoft, Tencent, and ByteDance have demonstrated the operational viability beyond reasonable doubt. What remains, and what this thesis has addressed, is the *engineering accessibility* of these systems: the ability for an operator outside the hyperscalers, with access only to public datasets, commodity hardware, and free Kaggle compute, to reproduce a competitive AIOps pipeline end-to-end and deploy it against their own fleet.

The Smart Storage AI pipeline demonstrates that this accessibility is now real. The 8.7 GB Alibaba dataset can be processed on a 16 GB workstation. The six-feature engineering set can be selected reproducibly. The per-architecture champions and the Universal Blind XGBoost can be trained on free GPU compute. The dashboard runs on a laptop. The path from public dataset to mission-control surface is, today, an engineering exercise of a few weeks, not a multi-year research programme.

The single most important empirical finding of this work — that a sparsity-aware Universal Blind model achieves near-parity with the per-architecture local champions while collapsing 96% of the MLOps overhead — is, if it generalises beyond the Alibaba corpus, a substantive

operational result. It means that the long-running engineering instinct of *one model per drive model* is not, in fact, optimal at hyperscale; that the additional MLOps cost of heterogeneity has been over-paid for years; and that the right architectural primitive (the XGBoost sparsity-aware split) has been sitting in the public open-source ecosystem since 2016, waiting to be applied to this exact problem.

This thesis is a step in that direction. The next steps — cross-dataset validation, federated training, online adaptation, and the universal sparsity-tolerant deep-learning model — are now open engineering problems with reasonably well-understood approaches. The author hopes that the open-source release of Smart Storage AI makes them slightly easier to address.

# Bibliography

- [1] Chakrabortii, C., and Boettner, R. (2023). Improving SSD reliability using an LSTM-based GAN for anomaly detection. *Proceedings of the 2023 IEEE International Conference on Big Data (BigData'23)*.
- [2] Chakrabortii, C., and Litz, H. (2020). Improving the accuracy, adaptability, and interpretability of SSD failure prediction models. *Proceedings of the 11th ACM Symposium on Cloud Computing (SoCC'20)*, 120–133.
- [3] Chen, T., and Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'16)*, 785–794.
- [4] Han, S., Lee, P. P. C., Xu, F., Liu, Y., He, C., and Liu, J. (2020). StreamDFP: A general stream mining framework for adaptive disk failure prediction. *Proceedings of the 49th International Conference on Parallel Processing (ICPP'20)*.
- [5] Han, S., Wu, J., Xu, E., He, C., Lee, P. P. C., Qiang, Y., Zheng, Q., Huang, T., Huang, Z., and Li, R. (2021). Toward reliable and efficient SSD storage systems: Lessons learned from real-world production data centers. *USENIX FAST'21*, 1217–1232. [Source of the Alibaba SSD failure dataset.]
- [6] Han, S., Lee, P. P. C., Liu, Y., and Wu, J. (2023). Distributed online stream mining for adaptive disk failure prediction. *IEEE Transactions on Parallel and Distributed Systems*, 34(7), 2106–2122.
- [7] Koh, C., Kang, J., Kim, T., and Han, S. W. (2024). Temporal-contextual attention network for solid-state drive failure prediction in data centers. *IEEE Access*, 12, 152398–152415.
- [8] Li, P., Dang, W., Lyu, C., Xie, M., Bao, Q., Ji, X., and Zhou, J. (2021). Reliability characterization and failure prediction of 3D TLC SSDs in large-scale storage systems. *IEEE Transactions on Device and Materials Reliability*, 21(2), 224–235.
- [9] Lu, S., Luo, B., Patel, T., Yao, Y., Tiwari, D., and Shi, W. (2020). Making disk failure predictions SMARTer! *Proceedings of the 18th USENIX Conference on File and Storage Technologies (FAST'20)*, 151–167.
- [10] Lukovic, V., Jovanovic, Ž., Djurasevic Pesovic, S., Pesovic, U., and Djordjevic, B. (2025). Solid-state drive failure prediction using anomaly detection. *Electronics*, 14(7), 1433.

- [11] Luo, C., Zhao, P., Qiao, B., Wu, Y., Zhang, H., Lu, W., Dang, Y., Lin, Q., and Zhang, D. (2021). NTAM: Neighborhood-temporal attention model for disk failure prediction in cloud platforms. *Proceedings of The Web Conference (WWW'21)*, 1181–1191.
- [12] Lyu, Y., Li, H., Sayagh, M., Jiang, Z. M., and Hassan, A. E. (2021). An empirical study of the impact of data splitting decisions on the performance of AIOps solutions. *ACM Transactions on Software Engineering and Methodology*, 30(4), 1–38.
- [13] Meza, J., Wu, Q., Kumar, S., and Mutlu, O. (2015). A large-scale study of flash memory failures in the field. *ACM SIGMETRICS Performance Evaluation Review*, 43(1), 177–190.
- [14] Pincioli, R., Yang, L., Alter, J., and Smirni, E. (2023). Lifespan and failures of SSDs and HDDs: Similarities, differences, and prediction models. *IEEE Transactions on Dependable and Secure Computing*, 20(1), 256–272.
- [15] Schroeder, B., Lagisetty, R., and Merchant, A. (2016). Flash reliability in production: The expected and the unexpected. *Proceedings of the 14th USENIX Conference on File and Storage Technologies (FAST'16)*, 67–80.
- [16] Xu, E., Zheng, M., Qin, F., Xu, Y., and Wu, J. (2019). Lessons and actions: What we learned from 10k SSD-related storage system failures. *Proceedings of the 2019 USENIX Annual Technical Conference (USENIX ATC'19)*, 961–976.
- [17] Xu, F., Han, S., Lee, P. P. C., Liu, Y., He, C., and Liu, J. (2021). General feature selection for failure prediction in large-scale SSD deployment. *Proceedings of the 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'21)*, 263–270.
- [18] Zhang, J., Zhou, K., Huang, P., He, X., Xiao, Z., Cheng, B., Ji, Y., and Wang, Y. (2019). Transfer learning based failure prediction for minority disks in large data centers of heterogeneous disk systems. *Proceedings of the 48th International Conference on Parallel Processing (ICPP'19)*.
- [19] Zhang, Y., Hao, W., Niu, B., Liu, K., Wang, S., Liu, N., He, X., Gwon, Y., and Koh, C. (2023). Multi-view feature-based SSD failure prediction: What, when, and why. *Proceedings of the 21st USENIX Conference on File and Storage Technologies (FAST'23)*, 409–424.
- [20] Zhang, Y., Zhang, T., Hao, W., Wang, S., Liu, N., He, X., Zhang, Y., Wang, W., Cheng, Y., Gwon, Y., Na, J., Kim, Z., and Oh, G. (2024). MSFRD: Mutation similarity based SSD failure rating and diagnosis for complex and volatile production environments. *Proceedings of the 2024 USENIX Annual Technical Conference (USENIX ATC'24)*, 869–884.