Cureus Journal of Computer Science
https://doi.org/10.7759/s44389-026-00334-1

Original Article

Automatic Workload Repository Snapshot-Based CPU Criticality Prediction in Oracle RAC Using Supervised Learning

Tejashkumar Patel [1], Dr. Harshal Arolkar [1]

1. *Faculty of Computer Application and Information Technology, GLS University, Ahmedabad, IND*

Received: August 19, 2026 | Review began: August 27, 2026 | Review ended: September 16, 2026 | Published: October 22, 2026

© **Copyright** 2026
This is an open access article distributed under the terms of the Creative Commons Attribution License CC-BY 4.0., which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Abstract

Resource management in Oracle database servers is one critical problem due to increased infrastructure complexity and dynamic workload conditions. Workload and core utilization prediction are crucial for efficient resource management decisions in Oracle databases. Existing solutions, such as those within Oracle Enterprise Manager (OEM), primarily focus on forecasting server resource usage (e.g., CPU) using Automatic Workload Repository (AWR) data that is available within database tables. A significant limitation for resource management is Database Administrator's (DBA's) inability to process data for CPU prediction if that historical information exists only in static exported formats, such as HTML reports, rather than accessible database tables. Physical or virtual machine level CPU utilization can be estimated for automated resource management decisions in Oracle database. However, it is not easy to differentiate Physical or Virtual CPU cores at software level due to the limitation of AWR on it. Using AWR datasets generated with AutoAWR, we propose an ML-based model for CPU-load prediction from historical data. For workload prediction, parsed AWR snapshots were labelled into CRITICAL (YES/NO) states and used to train multiple supervised classifiers (Logistic Regression, Random Forest, Gradient Boosting, XGBoost, and Support Vector Machines with a Radial Basis Function kernel. Model selection was performed using repeated stratified cross validation (5 folds, 10 repeats) on the training subset, and the selected model was then evaluated on held out validation and test splits. Gradient Boosting achieved the best mean cross validated F1 score and demonstrated strong hold out performance, indicating that minimally engineered AWR key metrics can reliably identify CPU critical states.

Categories: Performance Tuning and Optimization, Data Engineering, Machine Learning (ML)

Keywords: oracle rac, oracle awr, cpu critical, workload prediction, cpu resource planning, machine learning, awr csv performance prediction, oracle awr csv, oracle automation

Introduction

Modern Oracle database systems support diverse workloads, including mixed transactional and analytical processing [1]. This heterogeneity makes static CPU allocation strategies obsolete, as they fail to adapt to dynamic workload characteristics or meet strict Service Level Objectives (SLOs). The rise of cloud-native, multi-tenant architectures further complicates resource management by requiring tight resource sharing alongside strict performance guarantees [2, 3, 4].

Traditionally, Database Administrators (DBAs) plan CPU resources using centralized graphical tools such as Oracle Enterprise Manager (OEM) or custom SQL queries to scan historical Automatic Workload Repository (AWR) records. Although OEM highlights basic CPU issues visually, it depends heavily on database performance data that must be

How to cite this article:
Patel T, Arolkar D (October 22, 2026) Automatic Workload Repository Snapshot-Based CPU Criticality Prediction in Oracle RAC Using Supervised Learning. Cureus J Comput Sci 3 : es44389-026-00334-1. DOI https://doi.org/10.7759/s44389-026-00334-1

collected and interpreted from the available monitoring infrastructure. The core challenge lies in systematically parsing offline HTML performance reports across different systems to identify bottleneck timelines and optimize resource allocation [5, 6].

Supervised machine-learning algorithms including Random Forest (RF), Gradient Boosting (GB), Extreme Gradient Boosting (XGBoost), and Support Vector Machines (SVM) can learn to identify CPU saturation states from existing snapshot metrics [7, 8]. However, standard approaches typically treat this as a time-series regression task, which fails to capture the threshold-driven, state-based nature of real database saturation decisions. Furthermore, the extreme class imbalance between normal and critical states causes standard pipelines to overlook rare but vital high-load events [8, 9].

Despite advances in technologies in the era of artificial intelligence, no existing work successfully integrates machine-learning algorithms to address the unique challenges of CPU prediction in heterogeneous database environments of Oracle. Current methods suffer from three critical limitations. First, they lack an independent solution to extract fine-grained resource data from static HTML reports. Second, they fail to model dependencies across different database versions to provide context-aware solutions. Third, they cannot rapidly adapt to evolving workload combinations in live environments.

This study proves that supervised algorithms can recognize CPU saturation patterns from raw AWR snapshot data without requiring manual threshold calibration beyond the labeling phase. Our framework manually processes raw observations through redundancy removal, feature extraction, feature engineering, and threshold-based binary label creation to form a structured, supervised dataset. We then train and evaluate five classifiers: Logistic Regression (LR), RF, GB, XGBoost, and SVM with a Radial Basis Function kernel (SVM-RBF). We guided model selection through repeated stratified cross-validation, choosing to prioritize Recall for the critical states to protect production environments from costly, undetected performance failures. This provides a reproducible, practical framework for proactive capacity planning in Oracle RAC environments without requiring extra infrastructure.

Background and related work

The AWR within Oracle databases is an important asset for DBAs responsible for system health monitoring and resource planning [10]. Effective use of AWR requires experience, since performance data drawn from dynamic sessions, wait events, and key resource statistics must be interpreted and turned into practical actions for tuning and troubleshooting. DBAs study temporal performance snapshots to locate bottlenecks, identify inefficient SQL execution, and determine the exact events responsible for performance degradation. This human-driven approach uses standard metrics to understand how data, applications, and system capacity interact [6].

To complement this analytical depth, OEM provides a visual interface that greatly enhances the DBA's capacity for real-time diagnostics. Through its GUI, OEM translates AWR's raw data into interactive dashboards, trend analyses, and comparative reports, allowing DBAs to track the evolution of system load and swiftly pinpoint periods of peak resource contention. OEM facilitates not just immediate troubleshooting but also informed decision-making for tasks like capacity planning and pre-emptive resource allocation [2].

However, migrating historical data between databases for strategic analytics calls for proactive preparatory steps. Already-generated AWR reports are tailored strictly for manual performance evaluation in their original instance; they lack the flexibility for direct use in strategic forecasting across heterogeneous environments. Consequently, planning for such data migrations requires a deliberate architectural approach that aligns raw workload data with the target database's operational context, ensuring that the extracted insights remain both relevant and actionable for the team.

Artificial intelligence can be effectively applied by running machine-learning models directly on data extracted from these AWR reports. When performance data are converted into a machine-readable format like capturing indicators for prolonged high CPU usage, hidden performance patterns that standard audits miss are revealed. By organizing time-based metrics and related context into supervised learning datasets, machine-learning algorithms can identify relationships, track performance trends over time, and detect resource anomalies far more accurately than traditional rule-based programming methods.

How to cite this article:
Patel T, Arolkar D (October 22, 2026) Automatic Workload Repository Snapshot-Based CPU Criticality Prediction in Oracle RAC Using Supervised Learning. Cureus J Comput Sci 3 : es44389-026-00334-1. DOI https://doi.org/10.7759/s44389-026-00334-1

By combining statistical methods with machine-learning models, database systems can better predict performance changes and spot unusual behaviour. Instead of looking back only after a crash or major slowdown, administrators can identify rising performance stress or possible failures in advance from real data and changing workloads. This makes it easier to improve reliability and support a more automated approach to resource management in large-scale databases [2].

Literature review

This study positions itself within the broader context of data-driven performance modelling by focusing on supervised classification of CPU critical states derived from AWR historical snapshots. Building on established principles in machine-learning based system prediction, the current work seeks to evaluate how structured AWR metrics, combined with minimal temporal enhancements, can be leveraged to reliably predict CPU saturation events. The following sections review key contributions across machine-learning algorithm benchmarking, resource utilization forecasting, supervised prediction frameworks and CPU workload modelling, forming the conceptual foundation for the methodology adopted in this research.

Machine-learning has become a central component of data-driven prediction and intelligent decision-making across modern industries, with its effectiveness closely tied to the nature, structure, and quality of underlying datasets [8]. The study shows that supervised learning provides a strong framework for classification-based decision making when historical structured data are transformed into meaningful input versus output relationships. It also highlights that no single model performs optimally across all datasets, necessitating rigorous evaluation and algorithm selection tailored to the statistical properties of the available data. In real world applications, ranging from cybersecurity to IoT and business systems, supervised models have shown strong capability in forecasting system states and enabling timely automated decisions.

A study by Basu et al. [11] demonstrates how nonlinear time-series and machine-learning models can effectively forecast complex hydrological behaviours by transforming long-term observational data into structured supervised datasets and selecting influential lag-based features. Although applied to groundwater flooding, the methodological insight - capturing nonlinear temporal dependencies to improve predictive accuracy - is directly relevant to modelling CPU load behaviour from AWR-derived sequences.

Comparative studies across multiple machine-learning algorithms, such as the evaluation conducted by Patel et al. [12] shows that predictive accuracy is fundamentally constrained by dataset structure and feature relevance rather than algorithmic complexity. These findings reinforce the need for context-specific feature selection and a multi-algorithm benchmarking approach when modelling CPU state transitions in diverse workload environments.

Research on behavioural prediction in sequential decision environments further highlights the value of transforming raw temporal data into structured supervised learning signals. The work of Chincholkar and Pandit [13] shows how sequential state transitions can be modelled effectively using machine-learning techniques, reinforcing the importance of capturing temporal relationships in prediction tasks. These ideas are directly applicable to CPU workload dynamics in time-ordered AWR snapshots.

Studies in nonlinear system forecasting also indicate that deep learning architectures such as Long Short-Term Memory (LSTM) can perform well when datasets exhibit strong temporal dependencies and nonlinearity [14]. Although CPU workloads are often bursty and non-stationary rather than smoothly seasonal, this still underlines the importance of choosing an algorithm that matches the temporal nature of the target domain.

Standard machine-learning frameworks emphasise a clear workflow of data preprocessing, feature construction, model training, and evaluation, which provides a strong basis for supervised classification of system states [15]. These principles support the conversion of raw AWR snapshots into structured, learning-ready datasets.

Comparative analyses in photovoltaic forecasting show that supervised algorithm performance can vary widely across structured time-dependent datasets [16]. This supports the need for empirical benchmarking when selecting suitable algorithms for CPU prediction tasks driven by AWR-derived telemetry.

How to cite this article:
Patel T, Arolkar D (October 22, 2026) Automatic Workload Repository Snapshot-Based CPU Criticality Prediction in Oracle RAC Using Supervised Learning. Cureus J Comput Sci 3 : es44389-026-00334-1. DOI https://doi.org/10.7759/s44389-026-00334-1

Survey studies on multi-core architectures also show growing use of machine-learning for prediction and classification of runtime processor behaviour, with accuracy shaped strongly by workload characteristics and feature design [17]. This reinforces the need for comparative evaluation when modelling CPU-load prediction across different algorithms.

Research on enterprise infrastructure further shows that telemetry-driven forecasting methods, including ARIMA, Holt-Winters, and supervised machine-learning models, can predict resource saturation even in specialised hardware environments. Although this study focuses on database storage layers rather than direct CPU performance classification, it explicitly utilizes historical AWR metadata views [18]. Therefore, the main idea remains completely relevant: telemetry-based predictive modelling using AWR datasets can help anticipate CPU and memory bottlenecks across Oracle deployments.

Phase-aware learning frameworks for multi-core CPU workload forecasting show clear accuracy gains by modelling latent operational regimes and capturing abrupt changes in utilization behaviour. Lozano and Gerstlauer in their study report up to 22% improvement over a phase-unaware setup, highlighting the value of accounting for behavioural phase changes in CPU prediction [9]. This also reinforces the idea that CPU time-series data are structurally non-stationary and shaped by regime-dependent patterns.

Materials And Methods

The complete diagrammatic presentation of our activity methods is shown in Figure *1*, illustrating how the process begins with raw Oracle AWR HTML reports and ends with a definitive binary prediction using formatted data passed through machine-learning algorithms. For clearer classification, this workflow is divided into four main phases. Phase 1 focuses on data management, where the raw HTML inputs are parsed into a structured dataset. Phase 2 covers supervised dataset creation, where our threshold-based labeling rules are executed on the samples. Phase 3 establishes the machine-learning framework setup, handling the cross-validation benchmarking runs on the training data. Finally, Phase 4 defines the prediction verification protocol to evaluate model performance on the completely independent, held-out datasets.

The core task is snapshot-level binary classification of CPU critical states (CRITICAL=YES/NO) using AWR-derived workload indicators, rather than time-series forecasting. Stratified splitting ensures that the minority class proportion is preserved across all partitions, supporting stable model selection under strong class imbalance.

How to cite this article:
Patel T, Arolkar D (October 22, 2026) Automatic Workload Repository Snapshot-Based CPU Criticality Prediction in Oracle RAC Using Supervised Learning. Cureus J Comput Sci 3 : es44389-026-00334-1. DOI https://doi.org/10.7759/s44389-026-00334-1

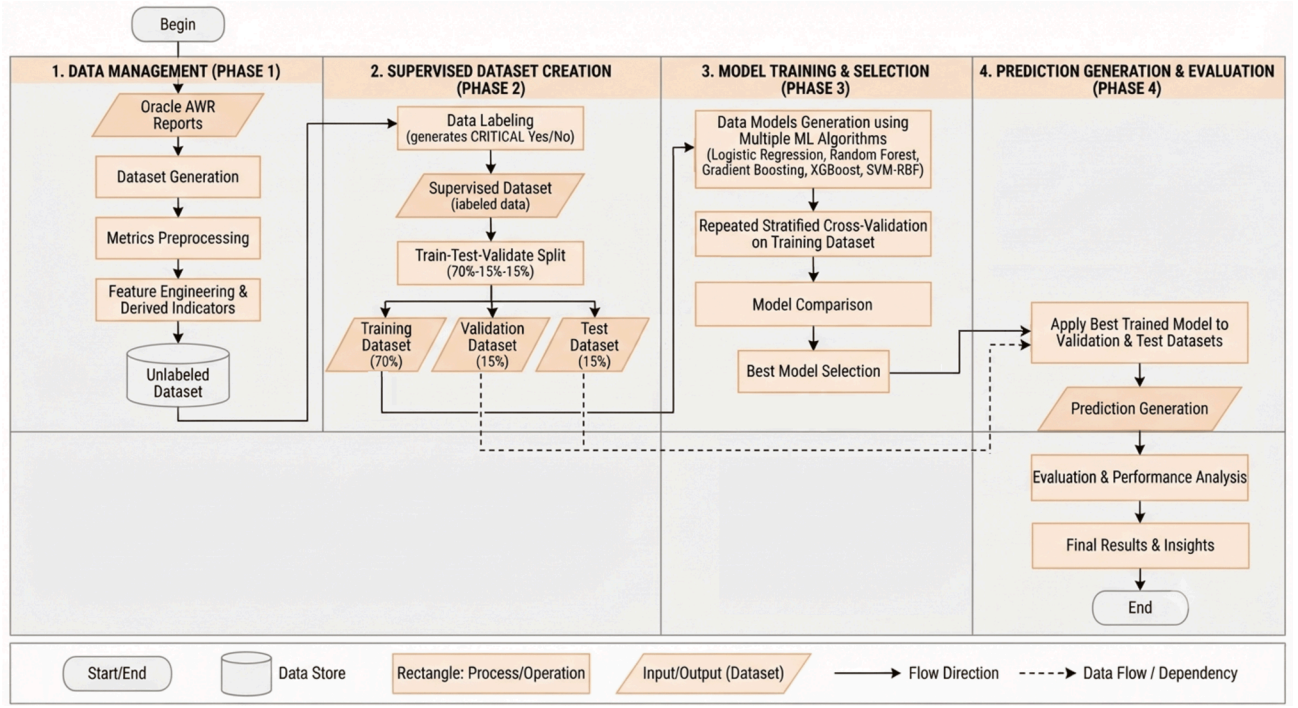

FIGURE 1: Complete procedural workflow architecture of the predictive machine-learning pipeline.

The detailed operational steps, activities, and logic rules executed within each of these four core phases are explained below.

Phase 1: Data management

The dataset was generated from historical Oracle AWR reports using AutoAWR, an automated data extraction and parsing platform that converted the reports into structured snapshot-level records with timestamps and workload metrics (Book chapter: Patel T, Arolkar H. AutoAWR: A Standardized Framework for Parsing and Structuring Oracle RAC AWR Reports into Analytical Datasets (IN PRESS). 5th World Conference on Information Systems for Business Management (ISBM 2026); 2026). Wherever available, we retained measures such as CPU utilization, Average Active Sessions (AAS), run-queue depth, and other OS-level indicators. Only metrics present across diverse Oracle deployments were kept to reduce environment-specific dependence, and the records were cleaned and standardised before feature construction.

The raw AWR data were cleaned by removing incomplete snapshots, maintenance spikes, and non-numeric values. Records were ordered chronologically, timestamps were standardised, and redundant or constant columns were removed to keep the dataset consistent.

We selected five AWR metrics as predictive features because they are directly related to CPU capacity, workload intensity, and host utilization. DB time was used as the core workload metric because it combines CPU time and non-idle wait time, while AAS provides a normalised activity rate across intervals. CPU Cores and %Busy capture host capacity and saturation: Performance Hub uses CPU Cores as the host reference line, and %Busy provides an OS-level proxy for non-idle CPU use. Together, these metrics support a compact and interpretable labelling strategy without time-series transformations. After preprocessing and feature extraction, the unlabelled snapshot records were passed to Phase 2 for labelling and supervised dataset creation.

How to cite this article:
Patel T, Arolkar D (October 22, 2026) Automatic Workload Repository Snapshot-Based CPU Criticality Prediction in Oracle RAC Using Supervised Learning. Cureus J Comput Sci 3 : es44389-026-00334-1. DOI https://doi.org/10.7759/s44389-026-00334-1

Phase 2: Three-way split and supervised dataset creation

The target label was generated using threshold-based criteria derived from CPU capacity, workload intensity, and host utilization. The labelling rules combine DB time, AAS, CPU Cores, and %Busy to identify intervals that are likely to be CPU saturated while keeping the feature set minimal and interpretable. Specifically, a snapshot was marked critical when it met one or more of three rules: a capacity rule based on AAS relative to CPU_Cores, a host-busy rule when %Busy exceeded 80, and a CPU-dominance rule using study-specific thresholds on DB time and DB CPU. These thresholds were chosen for conservative early-warning labelling, while the underlying metric definitions follow Oracle guidance, as formulated in Equation 1:

$$CRITICAL = [(AASCPU = 1) \vee (BUSY80 = 1) \vee ((CPUDOM = 1) \wedge (EARLYCAP = 1))] \quad (1)$$

The Boolean components of Equation 1 are evaluated at each snapshot interval as follows:

AASCPU ($AASCPU = 1$) evaluates to true if the AAS value exceeds the number of allocated physical CPU cores ($AAS > CPU_Cores$). This signifies that the database has more concurrent active sessions demanding processing time than the hardware layer can natively run in parallel.

BUSY80 ($BUSY80 = 1$) evaluates to true if the host operating system's total processing utilization percentage (%Busy) is strictly greater than 80% (%Busy > 80). This captures localized infrastructure bottleneck states.

CPUDOM ($CPUDOM = 1$) evaluates to true if Database CPU time constitutes the majority of overall Database Time ($\frac{DBCPU}{DBTime} > 0.90$). This indicates that workload latency is purely CPU-bound rather than caused by network, disk I/O, or concurrency locks.

EARLYCAP (EARLYCAP = 1) evaluates to true if the workload exhibits early capacity saturation warnings based on a study-specific conservative threshold ($AAS \geq 0.75 \times CPU_Cores$). The logical OR operators mandate that reaching a threshold in any single operational vector triggers a positive classification ($CRITICAL = YES$).

Applying the labelling rules produced a strongly imbalanced dataset: $CRITICAL = YES$, 161 records (1.38%); $CRITICAL = NO$, 11,479 records (98.62%). This reflects the operational reality that CPU-critical states are rare in production Oracle environments.

Figure *2* shows the class-conditional distributions of the five selected AWR metrics and illustrates how $CRITICAL = YES$ and $CRITICAL = NO$ differ in feature space.

How to cite this article:
Patel T, Arolkar D (October 22, 2026) Automatic Workload Repository Snapshot-Based CPU Criticality Prediction in Oracle RAC Using Supervised Learning. Cureus J Comput Sci 3 : es44389-026-00334-1. DOI https://doi.org/10.7759/s44389-026-00334-1

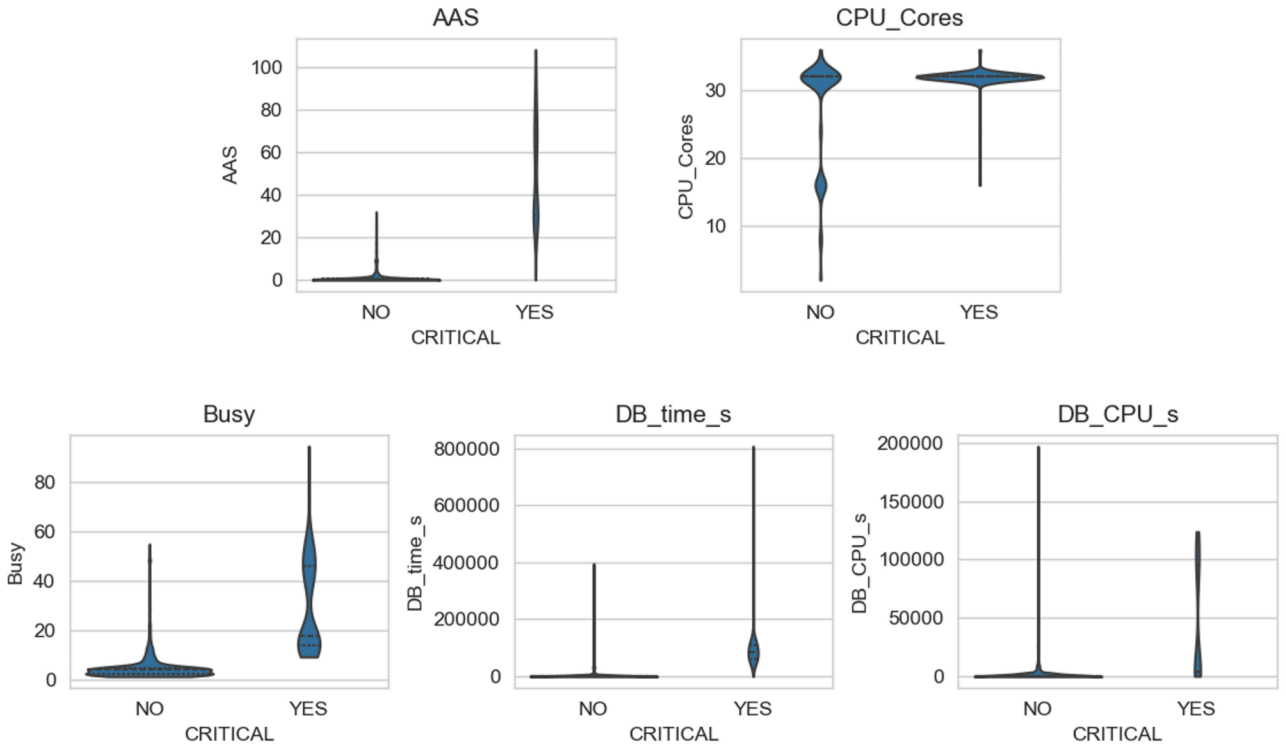

FIGURE 2: Distributions of the five AWR key metrics by class (CRITICAL=YES/NO).

AAS = Average Active Sessions, AWR = Automatic Workload Repository.

Each panel in Figure *2* corresponds to one metric: AAS, CPU Cores, %Busy, DB time (s), and DB CPU (s). These summary plots compare the distributions across classes to demonstrate that the selected features contain a meaningful signal for identifying critical CPU states. To accommodate distinct measurement units and operational scales, this grid layout utilizes independent vertical axis ranges for each of the five metrics. The vertical axes map the absolute metric distributions across the observed snapshots, reflecting raw concurrent counts for AAS ranging from 0 to 100, and static allocation capacities for CPU Cores ranging from 10 to 30. Host operating system strain (Busy) is tracked symmetrically on a standard percentage baseline from 0 to 100, whereas database execution latencies are scaled independently in absolute cumulative seconds, spanning up to 800,000 seconds for DB time (s) and up to 200,000 seconds for DB CPU (s). Segmenting these individual vertical scales ensures that structural distribution variances and class-conditional density shifts remain highly distinct and interpretable without numerical compression across parameters.

To properly isolate stable features from these historical metrics, it is essential to establish the underlying linearity and structure of the dataset. Traditional statistical benchmarks like the Pearson product-moment coefficient, Spearman's rank, and Kendall's Tau provide strong baseline insights but are mathematically restricted to parsing isolated, pairwise relationships between two variables only [19]. However, when analyzing enterprise architectures, workload factors do not change in isolation; rather, they interact as a complex multivariate ecosystem. In such environments, overlapping data structures create a high risk of multicollinearity, which can significantly destabilize model training weights, elevate standard errors, and severely distort feature importance indicators. Selecting optimized parameters therefore requires maximizing the retention of highly relevant variables while actively reducing redundant, similar variables across the feature space.

How to cite this article:
Patel T, Arolkar D (October 22, 2026) Automatic Workload Repository Snapshot-Based CPU Criticality Prediction in Oracle RAC Using Supervised Learning. Cureus J Comput Sci 3 : es44389-026-00334-1. DOI https://doi.org/10.7759/s44389-026-00334-1

Pearson correlation analysis was applied to the five continuous AWR metrics to quantify pairwise linear relationships and screen for early indicators of severe multicollinearity. This initial screening is required to guarantee that no two single operational parameters contain redundant variance that could destabilize model training weights or distort feature importance metrics. Pearson's correlation coefficient r is defined in Equation 2 below:

$$r = \frac{cov\,(X, Y)}{\sigma x \sigma y} \tag{2}$$

Where $cov(X, Y)$ denotes the covariance between the metric variables X and Y, while $\sigma x \sigma y$ represent the respective standard deviations of x and y. In this context, a coefficient value near +1 indicates a strong positive co-movement between workload metrics (such as active database processing directly elevating host-level CPU utilization), while a value near 0 implies a total linear independence between parameters (such as static physical hardware limits vs. transient active sessions).

Figure *3* shows the resulting Pearson correlation matrix heatmap.

How to cite this article:
Patel T, Arolkar D (October 22, 2026) Automatic Workload Repository Snapshot-Based CPU Criticality Prediction in Oracle RAC Using Supervised Learning. Cureus J Comput Sci 3 : es44389-026-00334-1. DOI https://doi.org/10.7759/s44389-026-00334-1

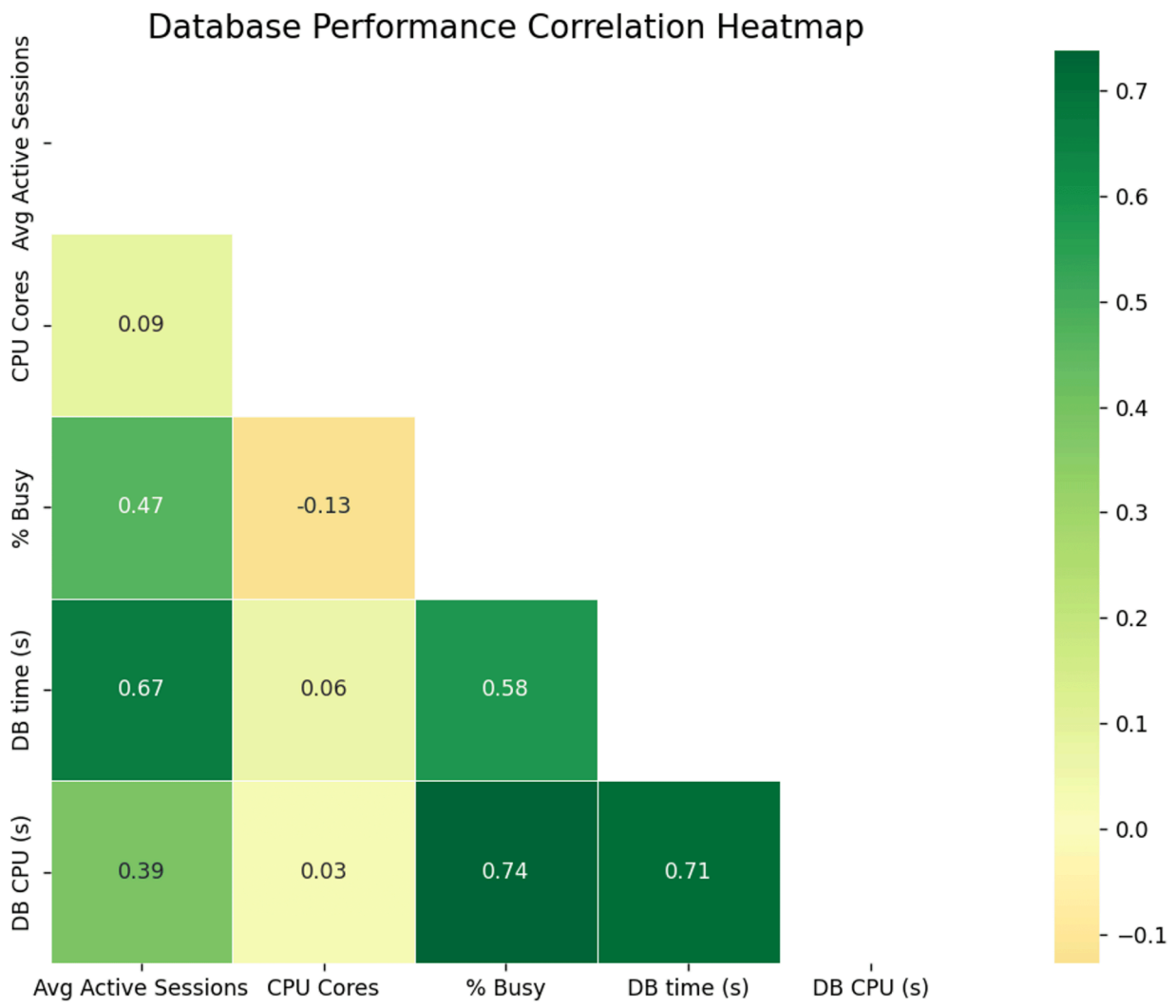

FIGURE 3: Pearson correlation matrix of the selected AWR key metrics.

As shown in the symmetric matrix layout of Figure *3*, both the horizontal column axis and the vertical row axis share an identical ordering of the five key AWR parameters to map their pairwise intersections. Each intersecting cell displays the calculated Pearson correlation coefficient r revealing strong positive associations between DB CPU (s) and %Busy ($r = 0.74$), and between DB CPU (s) and DB time (s) ($r = 0.71$), indicating that higher CPU service time directly coincides with higher host CPU utilization. DB time (s) is also positively correlated with AAS ($r = 0.67$) and %Busy ($r = 0.58$). Conversely, CPU Cores shows near-zero correlation with the other variables, confirming its role as a static capacity parameter rather than a dynamic workload measure. Crucially, as no pairwise correlation exceeds the critical multicollinearity threshold of 0.80 widely enforced in modern feature selection frameworks, all five metrics represent independent variances and are safely retained as features for subsequent machine-learning model training [20].

How to cite this article:
Patel T, Arolkar D (October 22, 2026) Automatic Workload Repository Snapshot-Based CPU Criticality Prediction in Oracle RAC Using Supervised Learning. Cureus J Comput Sci 3 : es44389-026-00334-1. DOI https://doi.org/10.7759/s44389-026-00334-1

Although assessing isolated pairwise correlations provides initial structural insights, verifying the absence of severe multi-variable dependencies is essential for effective variable selection in modern machine-learning workflows. As stated by Zaki et al., in their demonstration, when multiple explanatory features share excessive redundant variance, a condition known as multicollinearity arises. This statistical phenomenon artificially inflates estimator errors, destabilizes structural feature-importance metrics, and misleads optimization models into prioritizing redundant data patterns. To mitigate this issue, recent predictive frameworks utilize diagnostic filtering layers, such as the Variance Inflation Factor (VIF) to rigorously isolate independent variances from overlapping telemetry profiles. Proactively confirming structural feature independence ensures that the selected metrics contribute distinct operational value to the predictive model [21]. This systematic pruning directly enhances model generalizability, prevents algorithmic overfitting to transient spikes, and ultimately improves overall classification accuracy across changing server workloads.

To formally validate the absence of multicollinearity beyond pairwise correlations, the VIF was calculated for each independent AWR feature. The VIF for a given metric is defined in Equation 3.

$$VIF = \frac{1}{1 - R_i^2} \tag{3}$$

where R_i^2 denotes the coefficient of determination obtained by regressing the ith AWR metric against all remaining independent features in the workload dataset.

A VIF value of 1 indicates total independence, values between 1 and 5 signify moderate, manageable correlation, whereas values exceeding the strict conservative threshold of 5 indicate severe multicollinearity that requires feature removal, a boundary standard in contemporary predictive model optimization [21]. However, in purely predictive machine-learning frameworks, elevated correlation vectors do not inherently warrant the exclusion of features if each parameter retains distinct operational variance critical to mapping system anomalies [22]. All five selected AWR metrics yielded VIF scores well below 5, mathematically confirming that the feature set is completely stable and optimized for machine-learning model training, as visually demonstrated by projecting the calculated indices onto the baseline validation domain (Figure *4*).

How to cite this article:
Patel T, Arolkar D (October 22, 2026) Automatic Workload Repository Snapshot-Based CPU Criticality Prediction in Oracle RAC Using Supervised Learning. Cureus J Comput Sci 3 : es44389-026-00334-1. DOI https://doi.org/10.7759/s44389-026-00334-1

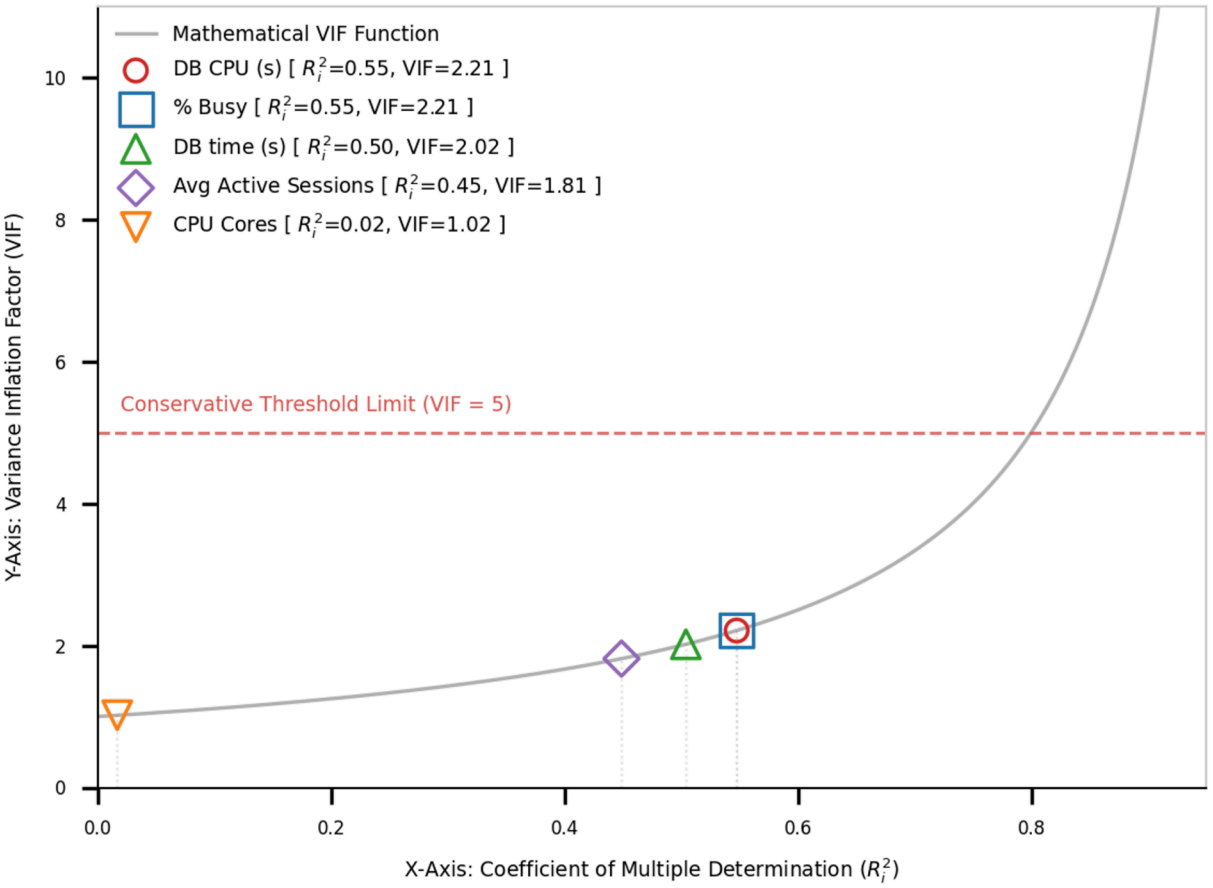

FIGURE 4: Variance inflation factor diagnostic function validating dataset feature independence

The baseline mathematical curve illustrated in Figure *4* tracks the exponential escalation of variance inflation trajectories across an increasing coefficient of multiple determination R_i^2. Calculated operational indices for all five core AWR parameters sit safely below the conservative threshold frontier of 5.0, confirming the structural independence of the feature space. This structural stability within the training dataset directly prevents the machine-learning model from overfitting to redundant architectural patterns. By ensuring that each active workload repository metric provides sufficiently distinct variance, the underlying mathematical risk of numerical instability during model optimization is minimized. Consequently, as the training matrix explicitly captures historical operational anomalies, the model demonstrates robust generalization capabilities. During subsequent framework execution on production-scale workloads, the predictive engine maintains highly reliable inference performance even when encountering transient operational workload spikes or sudden concurrency anomalies.

Phase 3: Machine-learning framework setup

Following the verification of feature independence via VIF analysis and aligning with open-science recommendations to preserve features contributing to predictive variance, the prepared snapshots were compiled into a supervised training matrix [22]. A completeness check was executed to confirm zero missing values across the selected feature vectors.

How to cite this article:
Patel T, Arolkar D (October 22, 2026) Automatic Workload Repository Snapshot-Based CPU Criticality Prediction in Oracle RAC Using Supervised Learning. Cureus J Comput Sci 3 : es44389-026-00334-1. DOI https://doi.org/10.7759/s44389-026-00334-1

To maintain strict evaluation integrity, the dataset was partitioned using a stratified split into training (70%), validation (15%), and testing (15%) subsets, thereby preserving the identical minority class proportion across all three cohorts. Five candidate classification models were selected for evaluation: Logistic Regression, Random Forest, Gradient Boosting, XGBoost, and SVM-RBF. Model selection and hyperparameter optimization were performed using repeated stratified k-fold cross-validation (5 folds, 10 repeats) restricted solely to the training cohort, utilizing the mean cross-validated F1-score as the primary selection criterion.

Phase 4: Prediction generation and evaluation

The final phase of the framework establishes a systematic verification protocol to evaluate model generalization on unseen data and interpret the final decision logic. This phase is structured across four main methodological steps.

First, for out-of-sample execution, the finalized, trained classification models are applied independently to the completely independent, held-out validation and testing subsets. This step evaluates true model stability by ensuring that no data leakage from the training phase influences the final target predictions.

Second, a multi-diagnostic evaluation matrix is utilized where operational performance is quantified across a standard matrix of classification metrics. Overall accuracy, precision, recall, and F1-score are calculated from the resulting confusion matrix tracks, whereas the receiver operating characteristic-area under the curve (ROC-AUC) is computed to evaluate model separability across varying classification thresholds.

Third, an operational optimization bias is enforced within the evaluation matrix, designating recall for the minority CRITICAL=YES class as the primary optimization metric. This targeted optimization reflects the high engineering and business costs associated with a false negative, where a failure to alert can lead to unmitigated production system degradation.

Finally, *post hoc* model interpretation is conducted by executing feature importance ranking and parameter coefficient analyses. This mathematical step tracks the underlying global influence of the respective AWR metrics on final model decisions, verifying how the trained algorithm weights the interactions between dynamic workload sessions and absolute host capacity parameters.

Results And Discussion

Experimental setup

The machine-learning training pipeline and feature extraction workflows were executed on a dedicated analytics workstation. The hardware environment consisted of an Intel Core i7-8550U processor (4 cores, 8 threads, clock speed up to 1.8 GHz) paired with 16 GB of DDR4 RAM running at 2400 MHz. The underlying software stack was built on a 64-bit Windows 10 operating system. The interface architecture and core report parsing engine were implemented via PHP within the AutoAWR framework, while dataset partitioning, multicollinearity testing, and classification training workflows were executed using Python 3.10.

Dataset metrics and partitioning outcomes

Compiling the processed AWR snapshots into the master supervised training matrix yielded a final baseline dataset comprising 11,640 total operational samples. The data distribution exhibited severe class imbalance, with 161 records (1.38%) labelled CRITICAL=YES and 11,479 records (98.62%) labelled CRITICAL=NO.

Executing the stratified 70/15/15 partitioning protocol on this population produced a training subset of 8,148 samples, a validation subset of 1,746 samples, and an independent testing subset of 1,746 samples. The finalized trained classification model was applied independently to these held-out validation and test subsets to generate definitive target predictions.

How to cite this article:
Patel T, Arolkar D (October 22, 2026) Automatic Workload Repository Snapshot-Based CPU Criticality Prediction in Oracle RAC Using Supervised Learning. Cureus J Comput Sci 3 : es44389-026-00334-1. DOI https://doi.org/10.7759/s44389-026-00334-1

Model benchmarking

The machine-learning pipeline results are reported in two stages. First, Table *1* shows the benchmark comparison of all candidate classification algorithms using repeated stratified cross-validation on the training data. Second, Table *2* presents the final selected model's performance on the completely independent, held-out validation and test datasets.

Model	Cross-Validated Accuracy	Cross-Validated Precision	Cross-Validated Recall	Cross-Validated F1-Score	Cross-Validated ROC-AUC	Folds	Repeats
Gradient Boosting (GB)	0.9994	0.9834	0.9715	0.9767	0.9927	5	10
XGBoost (XGB)	0.9992	0.9804	0.9645	0.9715	0.9999	5	10
Random Forest (RF)	0.9991	0.9930	0.9433	0.9667	0.9981	5	10
Support Vector Machine (SVM-RBF)	0.9990	0.9528	0.9761	0.9635	1.0000	5	10
Logistic Regression (Log Reg)	0.9983	0.8956	0.9964	0.9424	1.0000	5	10

TABLE 1: Benchmark evaluation of candidate classifiers using repeated stratified cross-validation on the training data.

ROC-AUC = receiver operating characteristic-area under the curve.

Table *1* reports mean performance over repeated stratified 5-fold cross-validation repeated 10 times (50 train-validation evaluations). Gradient Boosting achieved the highest mean F1-score (0.9767), with strong precision (0.9834) and recall (0.9715), indicating a good balance between false alarms and missed critical events. XGBoost and Random Forest performed similarly, while SVM-RBF and Logistic Regression were slightly weaker. The consistently high ROC-AUC values (about 0.993-1.000) show strong separability between CRITICAL=YES and CRITICAL=NO.

How to cite this article:
Patel T, Arolkar D (October 22, 2026) Automatic Workload Repository Snapshot-Based CPU Criticality Prediction in Oracle RAC Using Supervised Learning. Cureus J Comput Sci 3 : es44389-026-00334-1. DOI https://doi.org/10.7759/s44389-026-00334-1

Evaluation Split	Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC	True Negatives (TN)	False Positives (FP)	False Negatives (FN)	True Positives (TP)
Validation (VAL)	GB	1.0000	1.0000	1.0000	1.0000	1.0000	1,722	0	0	24
Testing (TEST)	GB	0.9994	1.0000	0.9583	0.9787	0.9999	1,722	0	1	23

TABLE 2: Hold-out datasets validation and test performance of the selected GB model.

GB = gradient boosting; ROC-AUC = receiver operating characteristic-area under the curve.

Table *2* shows that the selected Gradient Boosting model evaluated a total of 24 positive cases in each split. The model achieved perfect separation on the validation split (24 TP, 0 FN) and only one missed critical event on the test split (23 TP, 1 FN = 1), with zero false positives. This yielded Recall = 0.9583 and F1 = 0.9787 on the test set. The model is therefore conservative in raising alerts while still capturing most critical states, which is desirable in operational monitoring.

Overall, the repeated cross-validation results and the hold-out validation/test results indicate that the selected model generalises well to unseen snapshots. The error pattern suggests a strong precision bias without a major loss in sensitivity, which helps reduce alert fatigue. Because the positive class is small in the hold-out sets, future work should validate the model over longer production periods and across changing workload regimes.

Limitations

This study has several limitations. It was conducted on Oracle Database 11g (11.2.0.3.0), 12c (12.1.0.2.0), and 19c (19.0.0.0.0) RAC environments, so the findings may not generalise to single-instance deployments, non-RAC architectures, or other database platforms such as PostgreSQL, MySQL, or SQL Server. The implementation is limited to Oracle RAC AWR files for controlled public validation, and the experiments do not fully characterise runtime behaviour or scalability on larger enterprise servers or cloud infrastructure. Finally, the dataset is highly imbalanced, and the hold-out validation/test splits contain only 24 positive cases each, which adds uncertainty to recall estimates. Future work should use larger and more diverse AWR collections, longer time windows, and additional validation strategies.

Conclusions

This study shows that machine-learning-based prediction can enhance traditional CPU monitoring by identifying CRITICAL operating conditions from historical Oracle AWR snapshots. Gradient Boosting achieved the strongest mean cross-validated F1-score and strong hold-out performance, making it a good candidate for automated monitoring and alerting. From an operational standpoint, historical analysis using this framework allows DBAs to accurately diagnose underlying infrastructure problems and map hidden metric interactions. The pipeline relies on standard workload metrics, so DBAs can adapt this methodology to set up their own localized machine-learning models to predict impending CPU capacity issues proactively. Future work should broaden validation to other environments, test longer time windows and evolving workloads, and explore temporal features or time-series forecasting for earlier detection.

How to cite this article:
Patel T, Arolkar D (October 22, 2026) Automatic Workload Repository Snapshot-Based CPU Criticality Prediction in Oracle RAC Using Supervised Learning. Cureus J Comput Sci 3 : es44389-026-00334-1. DOI https://doi.org/10.7759/s44389-026-00334-1

Additional Information

Author Contributions

All authors have reviewed the final version to be published and agreed to be accountable for all aspects of the work.

Concept and design: Tejashkumar Patel, Dr. Harshal Arolkar

Acquisition, analysis, or interpretation of data: Tejashkumar Patel

Drafting of the manuscript: Tejashkumar Patel

Critical review of the manuscript for important intellectual content: Dr. Harshal Arolkar

Supervision: Dr. Harshal Arolkar

Disclosures

Human subjects: All authors have confirmed that this study did not involve human participants or tissue.

Animal subjects: All authors have confirmed that this study did not involve animal subjects or tissue.

Conflicts of interest: In compliance with the ICMJE uniform disclosure form, all authors declare the following:

- **Payment/services info:** All authors have declared that no financial support was received from any organization for the submitted work.
- **Financial relationships:** All authors have declared that they have no financial relationships at present or within the previous three years with any organizations that might have an interest in the submitted work.
- **Other relationships:** All authors have declared that there are no other relationships or activities that could appear to have influenced the submitted work.

Data Availability Statements

The operational code and Oracle AWR datasets used here are proprietary assets of Spurt Informatics Limited and its clients, protected by strict NDAs. To safeguard privacy, raw strings were redacted in images to keep headers intact. Although raw files cannot be made public, all anonymized summary metrics are fully documented in the article. If proof of originality is required by editors or reviewers, authors can demonstrate the same via live online screen sharing sessions upon request.

Acknowledgements

The authors thank the School of Doctoral Research and Innovation (SDRI), GLS University, Ahmedabad, for supporting this doctoral research framework. We also thank Spurt Informatics Limited for providing the testing platforms, computing environment, and server infrastructure necessary to generate the Oracle AWR datasets evaluated in this study.

References

1. Oracle: Oracle Database In-Memory Guide, 19c. (2026). Accessed: August 3, 2026: https://docs.oracle.com/en/database/oracle/oracle-database/19/inmem/.
2. Higginson A, Dediu M, Arsene O, et al.: Database workload capacity planning using time series analysis and machine learning. SIGMOD '20: Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data. 2020, 769-783. 10.1145/3318464.3386140
3. Ma L, Aken DV, Hefny A, et al.: Query-based workload forecasting for self-driving database management systems. SIGMOD '18: Proceedings of the 2018 International Conference on Management of Data. 2018, 631-645. 10.1145/3183713.3196908

4. Raza B, Kumar YJ, Malik AK, Anjum A, Faheem M: Performance prediction and adaptation for database management system workload using Case-Based Reasoning approach. Information Systems. 2018, 76:46-58. 10.1016/j.is.2018.04.005
5. Oracle: Using Automatic Workload Repository Warehouse for Generating Performance Reports. Oracle Database Performance Tuning Guide, 19c. (2026). Accessed: August 3, 2026: https://docs.oracle.com/en/database/oracle/oracle-database/19/tdppt/using-automatic-workload-repository-warehouse.html.
6. Harrison G: Oracle Performance Survival Guide: A Systematic Approach to Database Optimization. Pearson Education, Boston, MA; 2009.
7. C Ababei Cristinel, MG Moghaddam Milad Ghorbani: A survey of prediction and classification techniques in multicore processor systems. IEEE Transactions on Parallel and Distributed Systems. 2019, 30:1184-1200. 10.1109/tpds.2018.2878699
8. Sarker IH: Machine learning: algorithms, real-world applications and research directions. SN Computer Science. 2021, 2:160. 10.1007/s42979-021-00592-x
9. Lozano ESA, Gerstlauer A: Learning-based phase-aware multi-core CPU workload forecasting. ACM Transactions on Design Automation of Electronic Systems. 2022, 28:23. 10.1145/3564929
10. Oracle: Automatic Workload Repository (AWR). (2026). Accessed: August 3, 2026: https://docs.oracle.com/en-us/iaas/performance-hub/doc/awr-report-ui.html.
11. Basu B, Morrissey P, Gill LW: Application of nonlinear time series and machine learning algorithms for forecasting groundwater flooding in a lowland karst area. Water Resources Research. 2022, 58:e2021WR029576. 10.1029/2021wr029576
12. Patel E, Kushwaha DS: An integrated deep learning prediction approach for efficient modelling of host load patterns in cloud computing. Journal of Grid Computing. 2022, 21:5. 10.1007/s10723-022-09639-6
13. Chincholkar O, Pandit D: Machine learning algorithm for NPC in game development. High Technology Letters. 2022, 28:788-795.
14. Almosova A, Andresen N: Nonlinear inflation forecasting with recurrent neural networks. Journal of Forecasting. 2022, 42:240-259. 10.1002/for.2901
15. Theocharides S, Theristis M, Makrides G, Kynigos M, Spanias C, Georghiou GE: Comparative analysis of machine learning models for day-ahead photovoltaic power production forecasting. Energies. 2021, 14:1081. 10.3390/en14041081
16. Zhang Y, Li Q, Xin Y: Research on eight machine learning algorithms applicability on different characteristics data sets in medical classification tasks. Frontiers in Computational Neuroscience. 2024, 18:1345575. 10.3389/fncom.2024.1345575
17. Doshi R, Hiran KK, Jain RK, et al.: Machine Learning: Master Supervised and Unsupervised Learning Algorithms with Real Examples. BPB Publications, New Delhi, India; 2021.
18. Gollapudi R : Event-aware multi-layer storage risk forecasting for oracle database estates using HAPF. International Journal of Computational and Experimental Science and Engineering. 2024, 10:4309-4331. 10.22399/ijcesen.5183
19. Kasaraneni PP, Kumar YVP, Reddy GP: Multiple linear regression-based correlation analysis of various critical weather factors and solar energy generation in smart homes. Engineering Proceedings. 2025, 87:106. 10.3390/engproc2025087106
20. Shrestha N: Detecting multicollinearity in regression analysis. American Journal of Applied Mathematics and Statistics. 2020, 8:39-42. 10.12691/ajams-8-2-1
21. Zaki A, Métwalli A, Aly MH, Badawi WK: 5G and beyond: channel classification enhancement using VIF-driven preprocessing and machine learning. Electronics. 2023, 12:3496. 10.3390/electronics12163496
22. Addressing Multicollinearity. (2025). Accessed: August 13, 2026: https://library.virginia.edu/data/articles/addressing-multicollinearity.

How to cite this article:
Patel T, Arolkar D (October 22, 2026) Automatic Workload Repository Snapshot-Based CPU Criticality Prediction in Oracle RAC Using Supervised Learning. Cureus J Comput Sci 3 : es44389-026-00334-1. DOI https://doi.org/10.7759/s44389-026-00334-1