Intelligent ECU Load Management in Electric Vehicles Using a Gated Multi-Stage Machine Learning Framework

Mrs. Vaishali Mishra[1], Dr. Sonali Kadam[2]
Research Scholar, Department of Computer Science and Engineering,
Vishwakarma Institute of Information Technology, Pune, India[1]
PhD Supervisor, Department of Computer Science and Engineering, Vishwakarma Institute of Information Technology &
Professor, Bharati Vidyapeeth College of Engineering for Women, Pune, India[2]

***Abstract*—The growing adoption of software-defined and electrified vehicle architectures has significantly increased the computational burden on electronic control units, leading to dynamic and non-stationary load conditions that can compromise real-time performance and system reliability. Conventional ECU load-management strategies are largely static or address isolated aspects of the problem, such as overload prediction or energy optimization, without providing an end-to-end decision mechanism for runtime load redistribution. This study proposes a leakage-safe, three-stage intelligent ECU load-management model for electric vehicles that jointly performs overload detection, target ECU recommendation, and load-shift magnitude estimation within a gated architecture. The proposed model used ensemble and boosting-based machine learning models with task-specific feature design to prevent data leakage and reduce computational overhead through conditional execution. The performance of the proposed model is measured on a multi-feature ECU dataset characterized by non-stationary operational conditions and significant class imbalance between normal and overload states and addressed using stratified sampling and SMOTE-based augmentation. The proposed model obtained the overload detection rate F1-score of 0.916 and a ROC–AUC of 0.996, the target ECU recommendation obtained the accuracy of 0.935, and load-shift estimation, yielding an R^2 of 0.988 with low prediction error. This study also conducted the statistical test and ablation analysis, which observed that performance gains were consistent and attributable to key designs such as imbalance-aware learning, leakage control, and gated inference. The final results show that the proposed model is an effective and deployable solution for intelligent ECU load management in next-generation electric vehicles.**

Keywords*—Electric vehicles; electronic control units; intelligent load management; machine learning; overload detection; resource allocation

I. Introduction

Over the last few years, there has been a rapid transition toward electrified, connected, and software-defined vehicles that has transformed the automotive electronic system [1]. Electric vehicles (EVs) increasingly depend on a dense network of electronic control units (ECUs) to support advanced driver assistance systems (ADAS), battery management, power electronics, vehicle dynamics control, and connectivity services [2]. The E-vehicles are integrated and work under dozens of ECU and that execute the distinct control and sensing tasks under strict in real-time [3]. The ECU computational load is no longer static and dynamically changes in driving, environmental factors, battery state, thermal stress, and the concurrent execution of multiple software functions [4].

In practical automotive systems, ECU operation is governed by strict system-level constraints that directly impact reliability and safety [5]. The real-time control tasks in the ECU, such as braking, steering, and battery management, operate under tight latency bounds ranging from 10 ms to 100 ms; beyond that, deadline violations can degrade control performance. Moreover, ECU processing units operate near high utilization thresholds of 70-90% CPU usage that even transient load spikes can trigger overload conditions. Modern vehicles integrating 40-150 ECUs exacerbate this challenge, as concurrent task execution, inter-ECU communication, and shared resource contention introduce complex dependencies across subsystems [6]. The thermal constraints and battery state variations influence processing efficiency and communication delays over in-vehicle networks such as CAN that impact timely task execution. The tightly coupled constraints make static load allocation strategies insufficient and highlight the need for intelligent, adaptive, and real-time load-management mechanisms that allow the proactively addressed overload to preserve the system stability [7].

The existing research on ECU load management and energy optimization has explored several research directions, such as architectural consolidation, rule-based load shedding, and energy-aware scheduling [8]. More recently, machine learning (ML) techniques have been used for load forecasting, energy management, and demand response in vehicular and power systems [9]. However, the majority of existing models focused on isolated issues such as predicting load demand, optimizing charging schedules, or balancing energy consumption at the grid level. In contrast, end-to-end ECU-level load management inside the vehicle remains insufficiently addressed.

In particular, most studies address overload prediction or energy optimization independently, without determining where the load should be migrated once an overload is detected. Second, few works estimate how much computational load should be shifted to a target ECU in order to alleviate overload without introducing secondary bottlenecks. Third, many ML-based approaches rely heavily on utilization metrics such as CPU usage during training, which can introduce feature leakage, leading to inflated accuracy and reduced robustness during real-world deployment. This study proposes a three-stage intelligent ECU load-management model that integrates overload detection, target ECU recommendation, and load-shift magnitude estimation into a unified, leakage-aware decision pipeline. Unlike monolithic or single-step load-balancing strategies, the proposed approach decomposes the problem into three specialized learning tasks: 1) binary classification for overload detection, 2) target ECU recommendation, and 3) regression for load-shift amount estimation. This task decomposition enables each model to be optimized for its specific decision structure and maintain the coherent end-to-end operation.

The proposed model is based on the gated inference that computationally expensive downstream stages are executed only when an overload is detected. The overload detection acts as a decision gate, which targets ECU recommendations and load estimation, which are triggered exclusively under overload conditions. The proposed model reduces the average runtime overhead and aligns with the real-time constraints of ECU environments. The proposed model also used the leakage-safe feature strategy, without direct CPU and memory utilization during overload detection, and used it in subsequent stages where such information is causally valid. The performance of the proposed model was measured using several evaluation parameters and tested under ensemble and boosting-based learning models such as RF, CatBoost, LightGBM, and Gradient Boosting. Statistical testing and ablation studies were also conducted to verify that the model performance improvements were observed in this study. The main contributions of this work are summarized as follows:

A leakage-safe, three-stage intelligent ECU load-management model that jointly addresses overload detection, target ECU recommendation, and load-shift magnitude estimation within a unified structure.

A gated inference architecture that reduces runtime overhead by conditionally activate the downstream decision stages only when overload is detected.

To perform statistical significance testing and ablation analysis, providing rigorous validation of model performance and design choices in a safety-critical context.

The remainder of this study is organized as follows: Section II reviews related work on ECU system based on ML models. Section III presents the proposed three-stage intelligent ECU load-management model. Section IV discussed the experimental result analysis of models. It presents the statistical analysis and validates the results of models. Section V discussed the limitations, and practical considerations of the proposed model. Finally, Section VI concludes the study and outlines future research of the study.

II. Related Work

A. ECU Architectures and Load Management in Modern Vehicles

The electronic devices in E-vehicle are more sophisticated due to the automobile industry's rapid shift to software-driven and electric technologies. Modern E-vehicles contain 40 and 150 ECUs and each ECU is used as specific sensing, control, or actuation tasks, data flow management, latency reduction, and system scalability [10], [11], [12]. Kim et al. [13] and Cosman et al. [14] argued with growing the number of ECU, and coupled with real-time processing need in autonomous driving. There is necessitates intelligent load-management mechanisms to prevent overload and performance degradation.

Several studies suggested the architectural design from a one-ECU-one-function strategy toward centralized or domain-based ECU system to improve resource utilization and reduce complexity [15]. Hegde et al. [16] suggested the role of AUTOSAR-based software abstraction to enable the modular integration and efficient load balancing across ECU. Another functional safety discussed the use of multicore ECUs such as freedom-from-interference (FFI) mechanisms to reliable execution of concurrent tasks under ISO 26262 compliance [17]. The study suggested the necessity of adaptive and intelligent ECU load-management strategies in next-generation vehicles.

B. Machine Learning–Based Overload Prediction and Load Forecasting

The day-by-day machine learning (ML) techniques were used to predict load patterns and manage energy demand in vehicular and power systems. Sung et al. [18] suggested the ML-based load control model that used SVM to adjust capacity and obtained the notable reductions in peak power demand. Muhammad et al. [19] also suggested the effectiveness of SVM and genetic algorithms that predict the ECU load demands and to enable the proactive scheduling.

Another Bhatnagar et al. [20] and Luo et al. [21] also suggested the ensemble and deep neural model to predict short-term power consumption with high accuracy. The suggested model obtained the strong predictive performance that focused on grid-level or charging-station load forecasting rather than real-time ECU-level overload detection. Most if the existing models depends on utilization metrics without explicitly addressing feature leakage that reduces the predictive accuracy and limited deployment reliability in safety-critical environments.

C. Task Offloading, Resource Allocation, and Energy Management

Another border area of investigation in E-vehicle of task distribution and resource allocation that extensively studied is the energy management and electric charging. Wang et al. [22] proposed an electrical energy management mechanism that combine the dynamic current regulation with electrical load perception to improve battery utilization and alternator voltage stability. Mandepudi [23] also suggested the adaptive ML-based energy management model for hybrid electric vehicles that optimizes power distribution across the internal combustion engine and battery using real-time ECU feedback.

Gujjarlapudi et al. [24] and Balakumar et al. [25] suggested the ML-driven model to demand response and dynamic pricing strategies to address the peak loads in distribution networks affected by large-scale electric vehicle. The model shows the potential of energy and load optimization in grid-centric and does not address intra-vehicle task migration or ECU-to-ECU load redistribution under overload conditions.

Most of the existing studies address either overload prediction or energy optimization in isolation, without end-to-end model to determine the overload occurs, and also studies the load migration in ECU, and how much load should be shifted. Another studies investigated the feature leakage and imbalance handling. There is limited attention in statistical validation and ablation analysis are essential to demonstrate the robustness of ML models in safety-critical in E-vehicle.

This study proposes a leakage-aware, three-stage intelligent ECU load-management model that combined the overload detection, target ECU recommendation, and load-shift estimation within a unified strategy with experimental evaluation, statistical significance test, and ablation studies.

TABLE I. COMPARATIVE ANALYSIS OF EXISTING STUDIES WITH PROPOSED MODEL

Study	Application Domain	ML / Optimization Technique	Overload Detection	Target ECU Selection	Load-Shift Estimation	Leakage-Aware Design	Statistical Validation
Kim et al. (2023) [13]	In-vehicle ECU load factors	Analytical / system analysis	✗	✗	✗	✗	✗
Trovao et al. (2019) [26]	EV energy & component integration	System-level modeling	✗	✗	✗	✗	✗
Willrett (2023) [27]	EV charging & grid integration	Communication-driven scheduling	✗	✗	✗	✗	✗
Cosman et al. (2024) [14]	Vehicle E/E architectures	Architectural analysis	✗	✗	✗	✗	✗
Plich et al. (2016) [28]	Battery & load diagnostics	Rule-based control	△	✗	✗	✗	✗
Wang et al. (2018) [22]	Electrical energy management	DCR + ELP strategy	△	✗	✗	✗	✗
Sung et al. (2015) [18]	Load scheduling	SVM-based prediction	△	✗	✗	✗	✗
Muhammad et al. (2023) [19]	ECU load forecasting	SVM, GA	✓	✗	✗	✗	✗
Bhatnagar et al. (2022) [20]	Power-grid load forecasting	LightGBM	✓	✗	✗	✗	✓
Kim M.H. et al. (2021) [29]	Multicore ECU scheduling	Task scheduling algorithms	✗	△	✗	✗	✗
Sugunaraj et al. (2020) [30]	ECU identification	KNN, DT, GNB	✗	✗	✗	✗	✓
Balakumar et al. (2024) [25]	EV charging demand response	DTR + XGBoost	✓	✗	✗	✗	✓
Luo et al. (2020) [21]	Charging-station load prediction	Stacked Autoencoder	✓	✗	✗	✗	✓
Proposed Work	**In-vehicle ECU load management**	**RF, CatBoost, LightGBM (gated pipeline)**	✓	✓	✓	✓	✓

(Note: ✓ = explicitly addressed; △ = partially addressed; ✗ = not addressed; ✓✓ = comprehensive (statistical tests + ablation))

Table I shows the comparative analysis of existing studies and the proposed model that indicate the prior works primarily addressed and isolated aspects such as load prediction, leakage management without providing a complete end-to-end solution. The proposed model uniquely integrates overload detection, target ECU recommendation, and load-shift estimation within a unified, leakage-aware strategy that shows the comprehensive statistical validation.

III. PROPOSED METHODOLOGY

This section presents the complete workflow of the proposed three-stage load management model. Fig. 1 shows the proposed three-stage intelligent ECU load-management model to designed operation in E-vehicle. The workflow starts with data preprocessing and feature engineering and transform into discriminative operational features. The proposed model consists of inference and execution module that is designed using three sequential and conditional triggered tasks. The feature subsets were used across tasks to leakage-free overload detection and enable the utilization-aware recommendation and regression Task. In Task A, RF model is designed to proactive overload detection; if no overload is detected, the system maintains normal operation to avoid unnecessary computational overhead. If the overload detected, Task B recommends the suitable target ECU for load migration using a CatBoost model then in Task C, measure the precise load-shift amount through a RF regressor. Finally, a feasibility and safety check enforces CPU headroom constraints before executing load migration using task offloading and resource reallocation. This gated and modular design shows that the robustness, computational efficiency, and deployment feasibility are suited for safety-critical and dynamically changing EV environments.

A. Dataset Description

It is important to note that the dataset used in this study is generated from simulated driving conditions designed to emulate realistic ECU behavior under various operational

scenarios. The simulation incorporates diverse factors such as vehicle dynamics, battery state, and environmental conditions, it does not represent direct telemetry collected from production electric vehicles. The dataset indicates the controlled assumptions about system behavior rather than fully captured the complexity, noise, and variability inherent in real-world ECU operations.

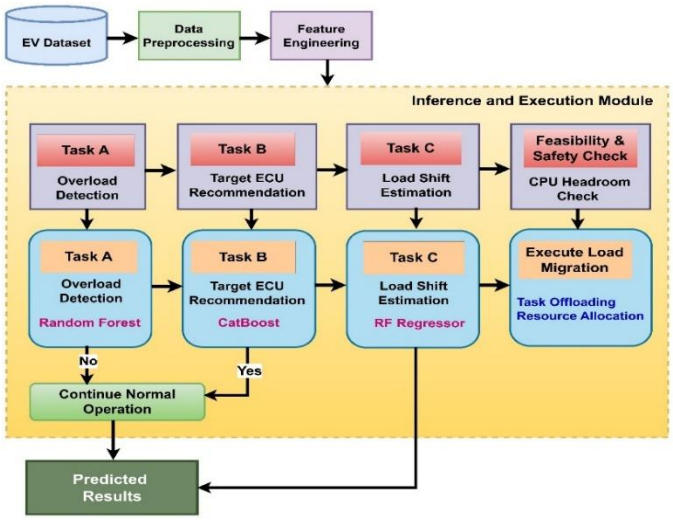


Fig. 1. Architecture of the proposed model.

B. Data Preprocessing and Imbalance Handling

This section maintains the data quality, prevents data leakage, and enable the reliable model training. The ECU dataset contains the several attributes such as ECU identifier, ECU function, task type, task priority, and charging status and continuous operational signals like temperatures, vehicle dynamics, and battery states. All samples in the dataset were examined to consistent and non-numeric categorical variables and were transformed into numerical form using label encoding. The raw dataset is denoted as $D = \{(x_i, y_i)\}_{i=1}^{N}$, where, $x_i \in R^d$ is the feature vector of each sample and y_i is the corresponding target variable. For categorical attributes, $c \in C$, to deterministic encoding function $f_{enc}(\cdot)$ was applied using Eq. (1):

$$x_{i,c}^{enc} = f_{enc}(x_i, c), \quad (1)$$

where, the mapping of each categorical value to an integer label is used to preserve the class separability.

The dataset was divided into training and testing subsets using the 80:20 splitting ratio. Due to the significant class imbalance between normal and overload conditions, synthetic oversampling was applied to the training data using the SMOTE. For a minority-class sample x_i, a synthetic instance x_{new} was generated, as indicated in Eq. (2):

$$x_{new} = x + \lambda(x_{nn} - x), \lambda \sim U(0,1) \quad (2)$$

where, x_{nn} is the k-nearest neighbors of x_i within the minority class. This process improves classifier sensitivity to overload events to avoid data leakage into the test set.

Fig. 2 shows the class distribution of the dataset before and after SMOTE. The original dataset shows severe class imbalance with the non-overload class (0) significantly dominating the overload class (1). After SMOTE, both classes are balanced to improved model learning and reduce the bias toward the majority class during training.

The visualization and certain learning stages, continuous features are standardized using z-score normalization [see Eq. (3)]:

$$x_{i,j}^{std} = \frac{x_{i,j} - \mu_j}{\sigma_j}, \quad (3)$$

Fig. 2. Distribution of the dataset before and after SMOTE.

where, μ_j and σ_j denote the mean and standard deviation of the j-th feature computed from the training data. The preprocessing steps of data integrity, leakage-free learning, and robust performance across all stages of the proposed ECU load-management model.

C. Exploratory Feature Space Visualization

To examine the intrinsic structure and discriminative capability of the selected input features prior to model training, an exploratory feature space visualization was performed using both dimensionality reduction and pairwise distribution analysis. This step provides qualitative validation of feature separability between normal and overload operating states and supports the subsequent choice of non-linear learning models for intelligent ECU load management.

1) Principal component analysis–based visualization: Given the high dimensionality of the engineered feature space, PCA was employed to project the data into a lower-dimensional subspace while retaining maximum variance. Let $X \in R^{n \times d}$ denote the standardized feature matrix, where, n is the number of samples and d is the number of features. PCA computes an orthogonal transformation such that [see Eq. (4)]:

$$Z = XW \tag{4}$$

where, $W = [w_1, w_2, w_3]$, consists of the eigenvectors corresponding to the three largest eigenvalues of the covariance matrix $\Sigma = \frac{1}{n-1} X^\top X$. The resulting low-dimensional representation, $Z \in R^{n \times 3}$ to capture the dominant features in the dataset.

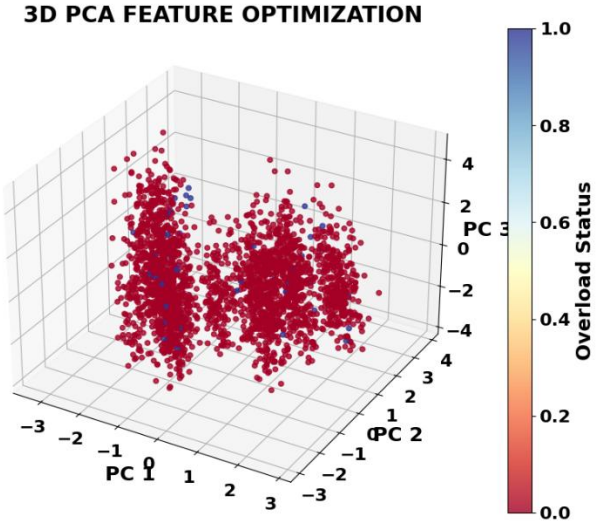


Fig. 3. 3D PCA feature optimization.

Fig. 3 shows the 3D PCA scatter plot, which was generated by adding samples onto the first three principal components and color-coding them according to overload status. Fig. 3 shows the partial but meaningful separation between normal and overload samples that shows the environmental and operational variables such as ECU temperature, battery state-of-charge, vehicle speed, and motor torque carry discriminative data in the absence of direct CPU usage features.

2) Pairwise feature distribution analysis: The pairwise feature distribution analysis was conducted on a subset of key physical and operational features such as ECU temperature, battery state-of-charge, vehicle speed, and motor torque. For each feature pair (x_i, x_j), joint distribution was visualized along with marginal kernel density estimates to examine class-wise overlap and interaction patterns. The given feature pair shows the joint probability density conditioned on class label $y \in \{0,1\}$ is expressed as in Eq. (5):

$$p(x_i, x_j \mid y) = p(x_i \mid y) p(x_j \mid x_i, y) \tag{5}$$

Fig. 4 shows the shifts in feature distributions under overload conditions during high-load ECU operation. This pairwise plot shows the intuitive, physics-consistent evidence to predictive relevance of the selected features.

D. Implications for Model Design

The combination of the PCA and pairwise analysis shows that the feature space that contain the non-linear class separability and inter-feature dependencies not adequately captured the linear decision models. The proposed tree-based ensemble and gradient boosting model is well-suited to visualize and capture the meaningful features rather than spurious correlations for training the models.

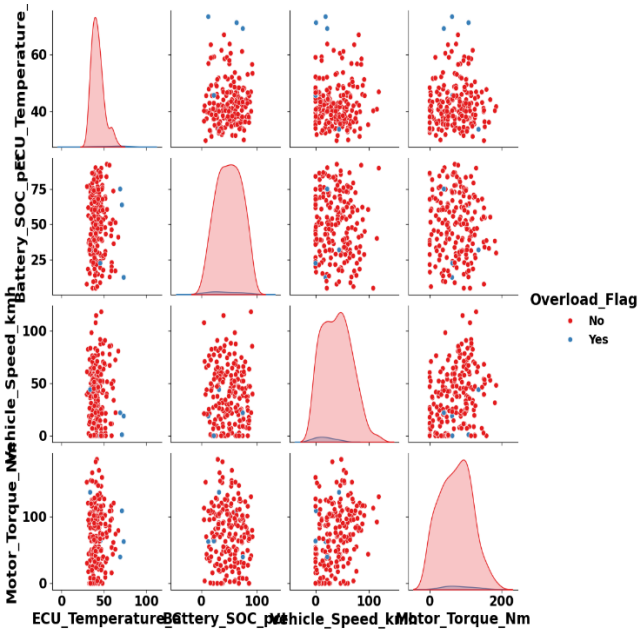


Fig. 4. Pairwise feature distribution analysis.

1) ECU monitoring model: Consider a set of ECUs $E = \{1,2,\dots,N\}$ denote the set of ECUs in the electric vehicle system, where, N is the total number of ECUs. At each discrete time instant t, for each ECU i at time t, a telemetry vector is observed [see Eq. (6)]:

$$x_{i,t} = [u_{i,t}, m_{i,t}, T_{i,t}, \dots]^\top \tag{6}$$

where, $u_{i,t}$ is CPU usage (%), $m_{i,t}$ is memory usage (%), and $T_{i,t}$ is ECU temperature (°C). Correlation analysis in the provided study indicates strong positive associations between CPU usage, memory usage, and temperature, consistent with physical workload behavior.

2) Three-task decision variables: The load-management decision at (i,t) is decomposed into:

a) Stage A: Overload Detection: The Stage A is to predict whether ECU_i will experience overload at time t. To prevent information leakage, direct CPU and memory utilization variables are excluded at this stage. The leakage-safe feature vector is defined as binary overload indicator [see Eq. (7)]:

$$\phi_{i,t}^{(A)} = g_A(x_{i,t}), \tag{7}$$

where, $g_A(\cdot)$ is a feature-selection operator that removes utilization-derived attributes. The overload detection model $f_A(\cdot)$ outputs the probability of overload [see Eq. (8)]:

$$p_{i,t} = f_A(\phi_{i,t}^{(A)}), p_{i,t} \in [0,1]. \quad (8)$$

The binary overload decision $\hat{y}_{i,t}$ is obtained using a decision threshold τ:

where, $\hat{y}_{i,t} = 1$ indicates overload and $\hat{y}_{i,t} = 0$ denotes normal operation.

b) Stage B: Target ECU Recommendation: Stage B is conditionally executed only if overload is detected $\hat{y}_{i,t} = 1$. In this stage, utilization features are included to enable informed target selection. The feature vector is defined as Eq. (9):

$$\phi_{i,t}^{(B)} = gB(x_{i,t}) = \phi_{i,t}^{(A)} \cup \{u_{i,t}, m_{i,t}\}, \quad (9)$$

where, $u_{i,t}$ and $m_{i,t}$ denote CPU and memory utilization, respectively. The target ECU recommendation model $fB(\cdot)$ produces a probability distribution over candidate ECUs [see Eq. (10)]:

$$q_{i,t} = f_B(\phi_{i,t}^{(B)}), \quad (10)$$

where, $q_{i,t} = [q_{i,t}(1), q_{i,t}(2), \dots, q_{i,t}(N)]$ and $q_{i,t}(j)$ represents the likelihood of selecting ECU j as the migration target. The recommended target ECU is then [see Eq. (11)]:

$$\hat{j}_{i,t} = \arg\max_{j \in E} q_{i,t}^{(j)} \quad (11)$$

c) Stage C: Load-Shift Amount Estimation (Regression): Once the target ECU $\hat{j}_{i,t}$ is selected, Stage C estimates the magnitude of workload to be migrated. Using the same utilization-aware feature vector $\phi_{i,t}^{(B)}$ the load-shift regressor $fC(\cdot)$ predicts [see Eq. (12)]:

$$\hat{\Delta}_{i,t} = fC\left(\phi_{i,t}^{(B)}\right), \quad (12)$$

where, $\hat{\Delta}_{i,t}$ denotes the predicted load-shift amount expressed as a percentage of ECU processing capacity.

3) Feasibility and Safety Constraint Enforcement: To ensure safe operation, the predicted load shift is constrained by the available CPU headroom of the target ECU [see Eq. (13)]:

$$\Delta_{i,t} = min\left(\hat{\Delta}_{i,t}, h_{\hat{j}_{i,t}t}\right), \quad (13)$$

where, $h_{\hat{j}_{i,t}t}$ represents the instantaneous CPU headroom of ECU $\hat{j}_{i,t}$. If $\Delta_{i,t} \leq 0$, no load migration is executed.

Fig. 5 shows the feature-importance ranking obtained from the winning model for overload detection that shows the dominant factors of ECU overload behavior detection. The temperature is the influential feature that indicate the strong relation across the thermal stress and computational overload in EV system. The task type and ambient temperature also indicate the workload characteristics and external operating conditions is important in shaping ECU load dynamics. The Battery-related variables like battery temperature and state of charge, with motor torque and vehicle speed, shows the interaction between vehicle dynamics, energy management, and ECU processing demand. The identifier-level attributes such as ECU ID and ECU function, and control parameters like charging status and task priority shows the lower importance. Real-time operating and physical factors used as static variables of the overloading prediction.

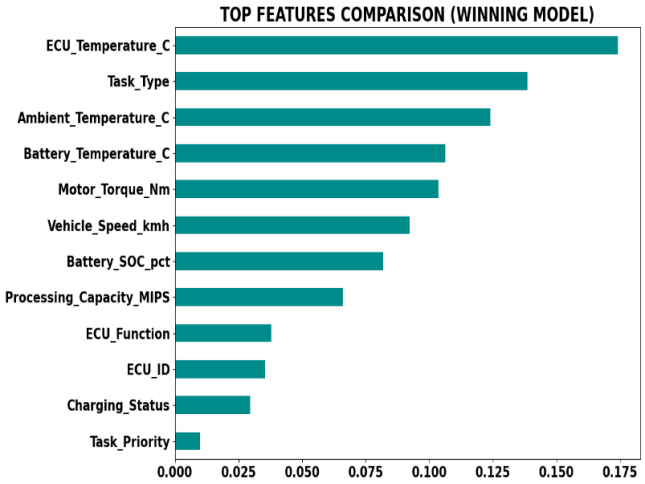


Fig. 5. Feature importance analysis of the winning overload detection model.

Algorithm 1: Three-Stage Intelligent ECU Load Management

```
Initialize trained models f_A, f_B, f_C
Initialize ECU set 𝔈 = {ECU_1, ECU_2, ..., ECU_n}
for each time step t do
    for each ECU i ∈ 𝔈 do
        Acquire telemetry vector X_{i,t}
        Perform preprocessing and feature encoding
        Stage A: Overload Detection
        Construct leakage-safe feature vector φ_A(X_{i,t})
        Predict overload probability p_A ← f_A(φ_A)
        if p_A < τ then
            continue  // No overload → skip to next ECU
        end if
        Stage B: Target ECU Recommendation
        Construct feature vector φ_B(X_{i,t})
        Predict target ECU j ← argmax f_B(φ_B)
        Stage C: Load Shift Amount Prediction
        Construct feature vector φ_C(X_{i,t})
        Predict load shift Δ ← f_C(φ_C)
        Feasibility & Safety Constraints
        Compute CPU headroom h_{j,t}
        Δ ← min(Δ, h_{j,t})
        if Δ ≤ 0 then
            continue  // Insufficient headroom → no migration
        end if
        Load Reallocation
        Migrate load Δ from ECU i to ECU_j
        Update ECU states and resource counters
    end for
end for
```

The above Algorithm 1 operates in a continuous monitoring loop, where ECU telemetry is processed at each time step. Stage A performs proactive overload detection using leakage-safe

features; if no overload is detected, downstream computation is skipped, low runtime overhead. If overload is detected the stage B recommends the suitable target ECU, and in stage C estimates the optimal load-shift magnitude.

4) Assumptions on ECU Independence and Interdependencies: The proposed models operates each ECU independently during inference to allow the computational efficiency and real-time feasibility. The overload detection, target recommendation, and load-shift estimation stages operate on per-ECU telemetry without construction of inter-ECU dependency graphs. This assumption simplifies the learning problem; it does not indicate that ECU operate in complete isolation.

In real-time systems, ECU shows the inherent interdependencies through shared communication networks, coordinated control tasks, and resource contention. In the proposed model interactions are captured through system-level features, such as vehicle dynamics such as speed, torque, battery state, and environmental conditions that indicate the aggregated influence of multiple ECU and subsystems. The utilization-based features used in later stages Task B and Task C indirectly encode system-wide load conditions.

The independence assumption enables scalable and low-latency inference, which is critical for real-time deployment in embedded environments. However, it represents a controlled abstraction of the underlying system dynamics. Future extensions of this work used explicit inter-ECU dependency modeling, such as graph-based learning, attention mechanisms to further enhance coordination across ECU.

IV. Result Analysis

This section presents the performance analysis of the proposed model. All simulations and model training were performed on Google Colab Pro environment with high-performance cloud infrastructure with GPU acceleration to stable execution of models. The dataset was partitioned using an 80:20 train–test split with stratification to preserve the original class distribution to imbalanced overload detection task. Task-specific hyperparameters were carefully selected based on empirical tuning and prior studies such as the number of estimators and tree depth for ensemble models, regularization strength for linear baselines, and kernel settings for support vector regression. Model performance was assessed using several evaluation parameters, as shown in Eq. (14) to Eq. (22). For overload detection (Task A) and target ECU recommendation (Task B), the following metrics are used. Let TP, TN, FP, and FN denote the number of true positives, true negatives, false positives, and false negatives, respectively.

$$Accuracy = \frac{T\_P + T\_N}{T\_P + T\_N + F\_P + F\_N} \quad (14)$$

$$Precision = \frac{T\_P}{T\_P + F\_P} \quad (15)$$

$$Recall = \frac{T\_P}{T\_P + F\_N} \quad (16)$$

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (17)$$

For Load Shift Amount Prediction (Task C), continuous-valued regression metrics are employed. Let y_i and $y\hat{}i$ denote the actual and predicted load-shift amounts for the i-th sample, and N be the total number of samples:

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i| \quad (18)$$

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \quad (19)$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2} \quad (20)$$

$$MAPE = \frac{100}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \quad (21)$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y}_i)^2} \quad (22)$$

TABLE II. Performance Analysis of Overload Detection of Task A.

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC
Random Forest	0.982	0.889	0.943	0.915	0.996
CatBoost	0.984	0.907	0.925	0.916	0.996
LightGBM	0.980	0.875	0.925	0.899	0.994
XGBoost	0.978	0.860	0.925	0.891	0.992
MLP Classifier	0.966	0.789	0.849	0.818	0.982
Logistic Regression	0.942	0.652	0.849	0.739	0.963

Table II shows the performance of overload-detection (Task A) in ECU across six classifiers, showing consistently high discrimination for the tree-based ensemble family. CatBoost obtained the best overall balance with Accuracy = 0.984, Precision = 0.907, Recall = 0.925, F1-score = 0.916, and ROC-AUC = 0.996, narrowly performed well RF obtained Accuracy = 0.982, Precision = 0.889, Recall = 0.943, F1-score = 0.915, ROC-AUC = 0.996 indicate the highest recall and thus stronger sensitivity to overload events. LightGBM and XGBoost shows the competitive but slightly lower in F1-score (0.899 and 0.891) and ROC-AUC (0.994 and 0.992). The neural baseline (MLP) degrades to F1 = 0.818 and ROC-AUC of 0.982 and LR shows the weakest precision of 0.652 and F1 of 0.739 that shows the limited capacity to capture the nonlinear overload patterns shows the moderate recall of 0.849.

TABLE III. Performance Analysis of Target ECU Recommendation of Task B.

Model	Accuracy	F1-Macro	Precision-Macro	Recall-Macro
CatBoost	0.935	0.933	0.932	0.934
Random Forest	0.913	0.911	0.912	0.911
LightGBM	0.913	0.911	0.912	0.911
XGBoost	0.891	0.888	0.889	0.889
MLP Classifier	0.870	0.868	0.871	0.866
Logistic Regression	0.826	0.823	0.830	0.824

Table III shows the performance of Target ECU Recommendation in Task B using macro-averaged parameters

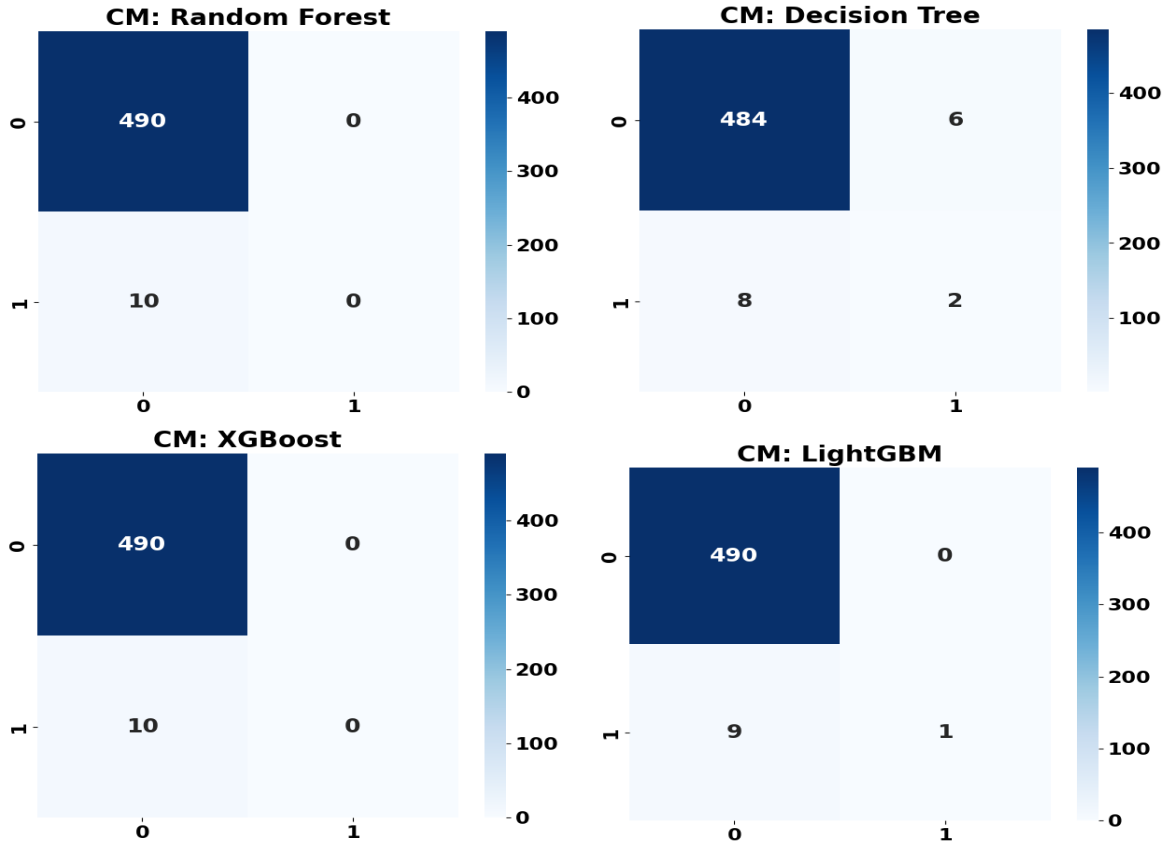
of proposed models across ECUs. CatBoost shows the strongest performance that obtained the accuracy of 0.935, F1-Macro = 0.933, Precision-Macro = 0.932, and Recall-Macro = 0.934 that indicate the both high correctness and consistent per-class reliability. The RF and LightGBM model are the next tier with same performance to obtained the accuracy of 0.913, F1-Macro = 0.911, Precision-Macro = 0.912, Recall-Macro = 0.91. The XGBoost shows the reduction of accuracy of 0.891, F1-Macro = 0.888. The MLP classifier obtained the accuracy of 0.870 and F1-Macro of 0.868 to reduced robustness for ECU-wise class boundaries. The LR is the weakest model that obtained the accuracy of 0.826, F1-Macro of 0.823 that indicate the limitations of linear decision surfaces for the heterogeneous workload in reliable target ECU selection.

Table IV presents the regression performance for Load Shift Amount Prediction (Task C), where ensemble tree models clearly dominate. The RF model obtained the accuracy with R^2 = 0.988, MAE = 1.424, MSE = 5.174, and RMSE = 2.275 that highly precise estimation of the required load migration with low average deviation. The LightGBM model obtained the R^2 of 0.988, MAE of 1.447, and RMSE of 2.315 then followed by the gradient XGB model of R^2 of 0.987, MAE of 1.513, RMSE of 2.372 and XGBoost obtained the R^2 of 0.987, MAE of 1.532, and RMSE of 2.398, that indicate the marginal degradation relative to RF model. The SVR model degrades performance to R^2 = 0.726 with RMSE = 8.014, and the LR model declines to R^2 = 0.687 with RMSE = 8.538, indicating that the non-linear ensemble learners' model is more effective to capturing the complex relationship between ECU telemetry and continuous load-shift quantities.

Fig. 6 shows the confusion matrix of the proposed model in Task A for overload detection that shows the class-wise prediction behavior under imbalanced conditions. The RF, XGBoost, and LR models correctly classify all 490 normal samples (TN = 490) but fail to identify overload events, misclassifying all 10 overload instances as normal (FN = 10, TP = 0) that indicate the extremely conservative behavior with zero false positives but poor overload sensitivity. The DT and Gradient XGB models show the improved balanced performance and each correctly detected 2 overload cases (TP = 2) and incurred the 6 false positives (FP = 6) and 8 false negatives (FN = 8) indicate the moderate sensitivity at the cost of increased false alarms. The LightGBM model obtained the better performance to correctly identify the 1 overload instance (TP = 1) with only 1 false positive (FP = 1) and 9 false negatives (FN = 9) instances to maintain the high specificity to slightly improve the recall over conservative models.

TABLE IV. PERFORMANCE ANALYSIS OF LOAD SHIFT AMOUNT PREDICTION IN ECU OF TASK C.

Model	R-Squared	MAE	MSE	RMSE
CatBoost	0.988	1.424	5.174	2.275
Random Forest	0.988	1.447	5.357	2.315
LightGBM	0.987	1.513	5.626	2.372
XGBoost	0.987	1.532	5.750	2.398
MLP Classifier	0.726	5.187	64.219	8.014
Logistic Regression	0.687	5.809	72.899	8.538



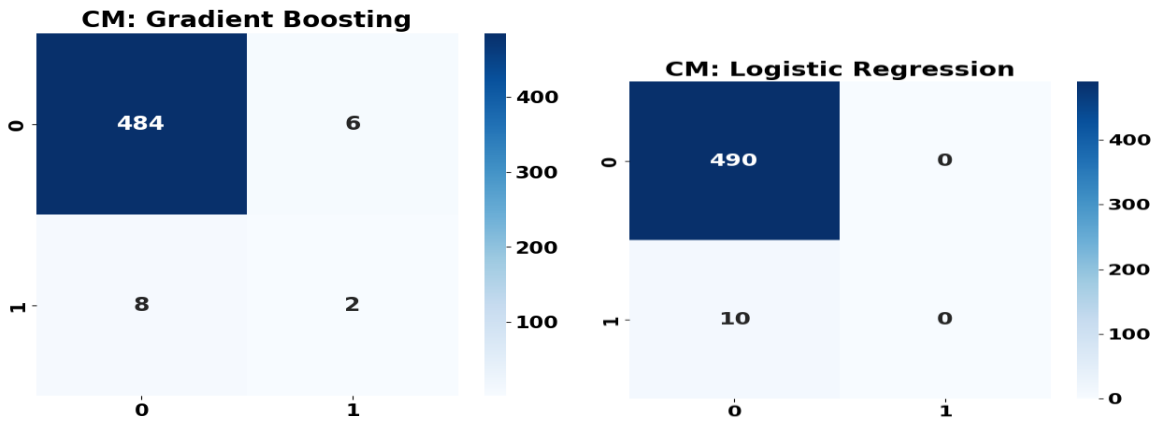


Fig. 6. Confusion matrix of proposed models for overload detection of Task A.

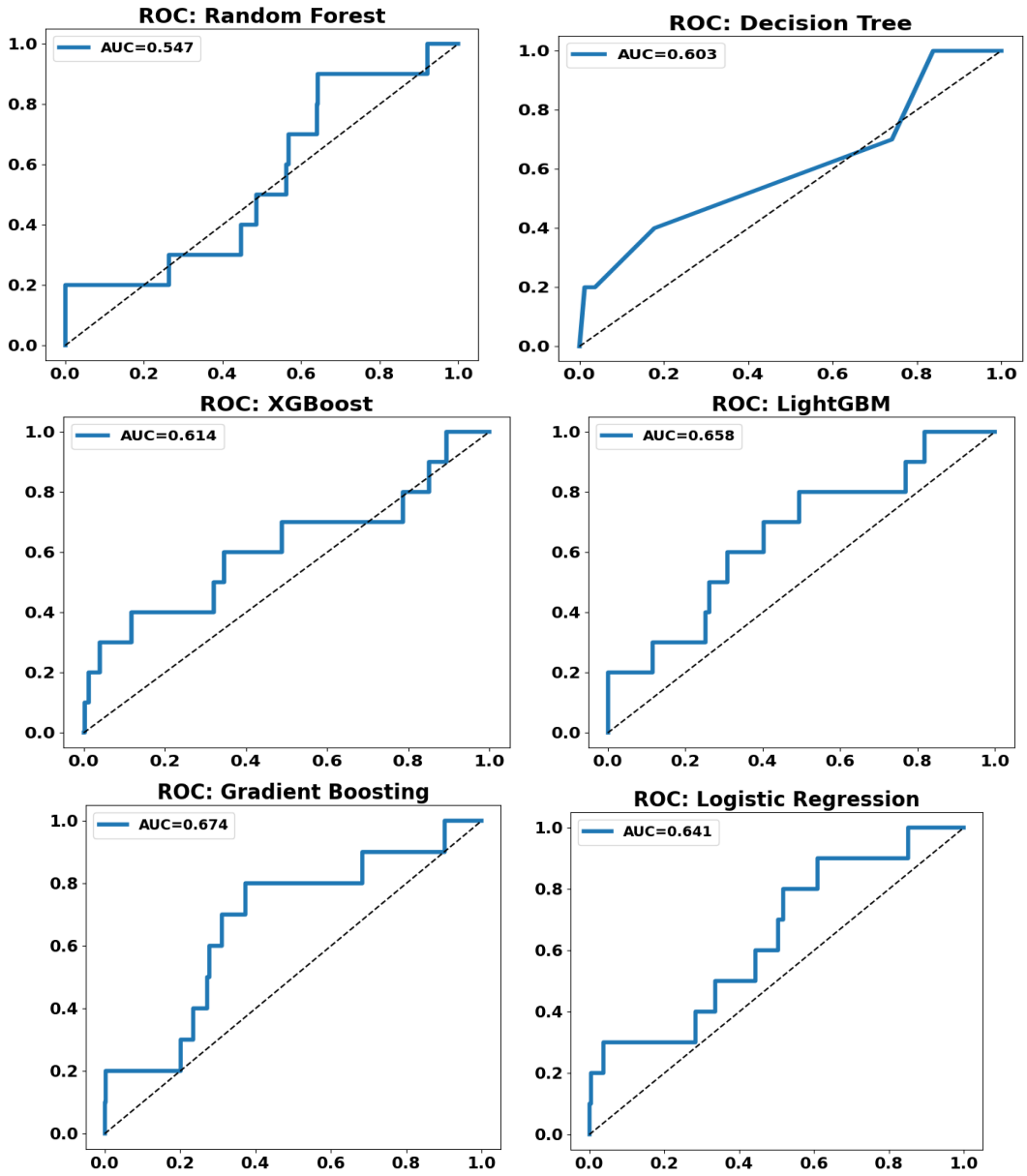
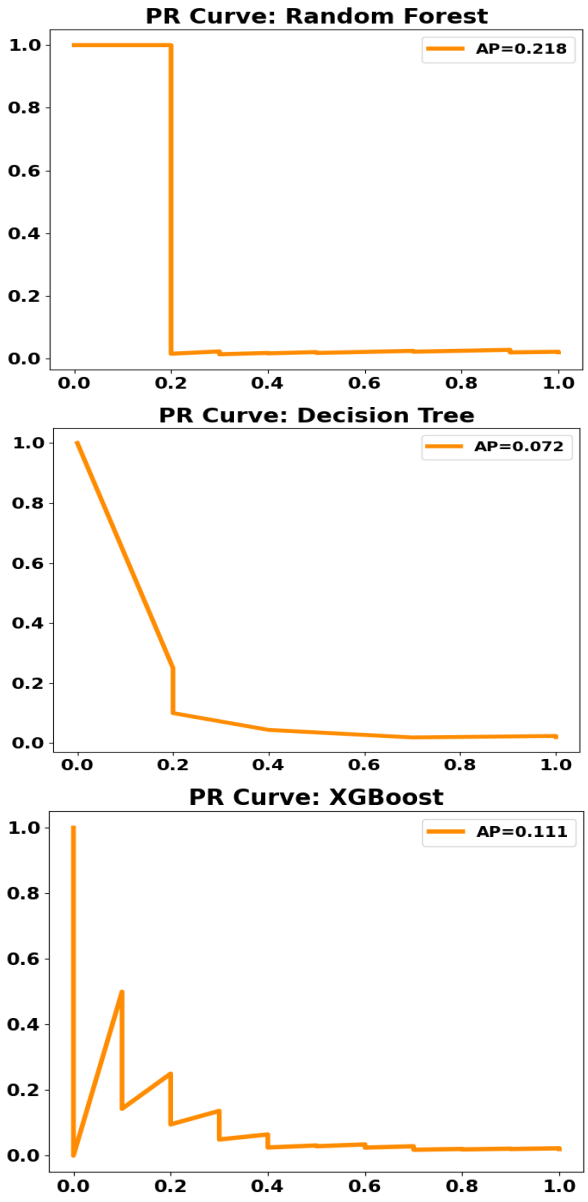


Fig. 7. ROC curve of proposed models for overload detection of Task A.

Fig. 7 shows the ROC curves for the evaluated models in Task A (Overload Detection) that indicate the ability to discriminate between normal and overload operating states under class imbalance. Among the models, Gradient Boosting achieves the highest discrimination capability with an AUC of 0.674, followed by LightGBM (AUC = 0.658) and Logistic Regression (AUC = 0.641), indicating comparatively better separation between positive and negative classes. XGBoost and Decision Tree attain moderate performance with AUC values of 0.614 and 0.603, respectively, while Random Forest shows near-random discrimination with an AUC of 0.547, reflecting its conservative decision boundary under the selected operating threshold.



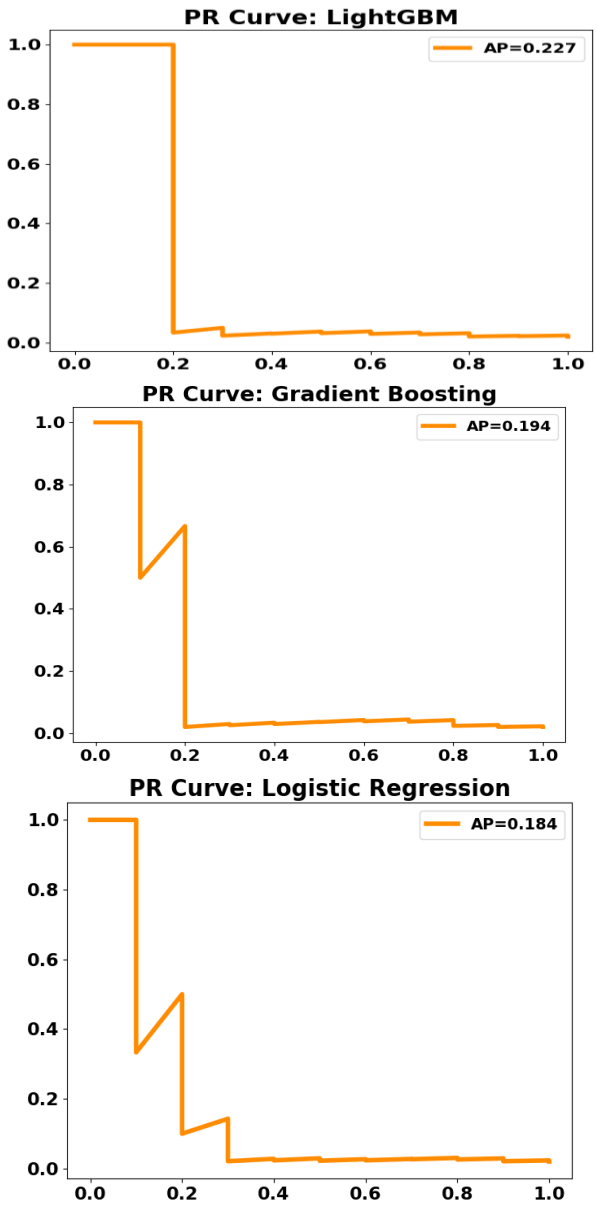


Fig. 8. PR curves of proposed models for overload detection of Task A

Fig. 8 shows the PR curves of proposed models in Task A that indicate the class-imbalance–aware performance of positive class identification. The LightGBM obtained the highest average precision with AP = 0.227, then RF obtained the AP of 0.218 and Gradient XGB obtained AP of 0.194, that indicate the comparatively better PR trade-offs to detect the overload events. The LR model obtained the moderate performance AP of 0.184 and XGBoost and DT achieved an AP = 0.111 and 0.072, respectively that indicate the limited precision at higher recall levels.

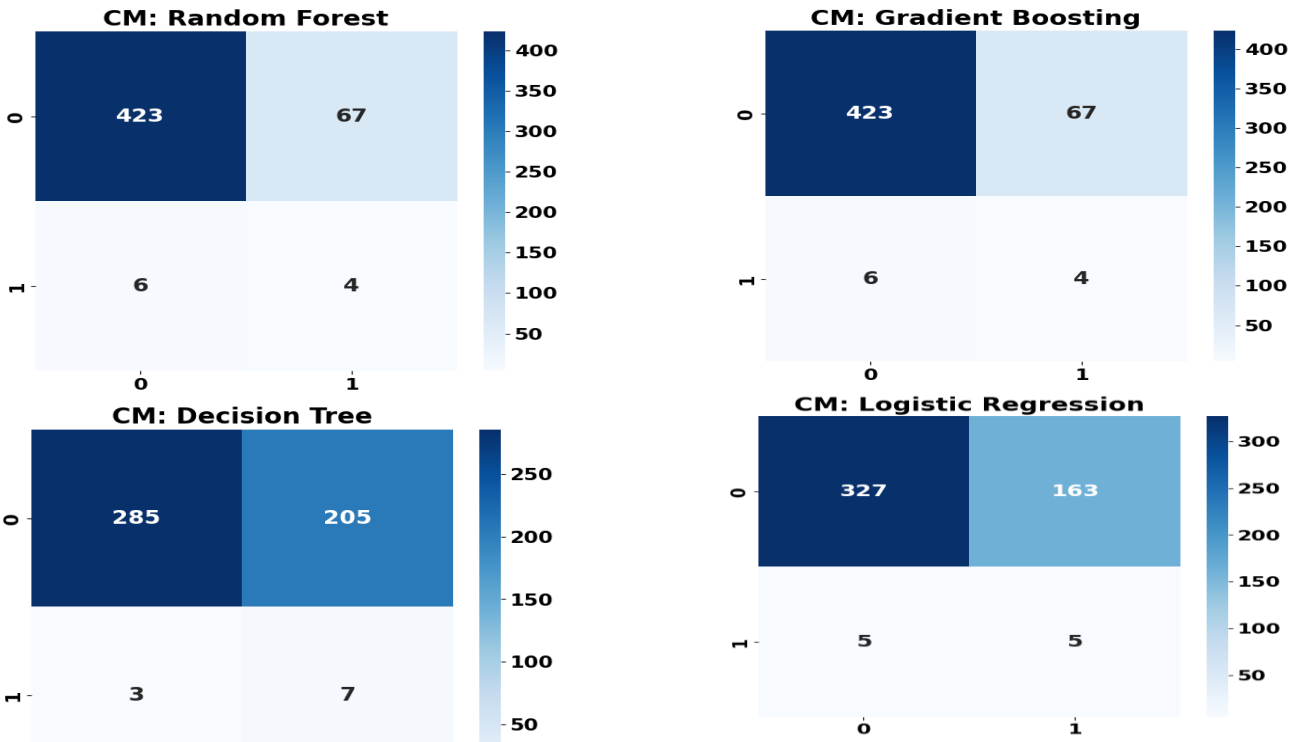
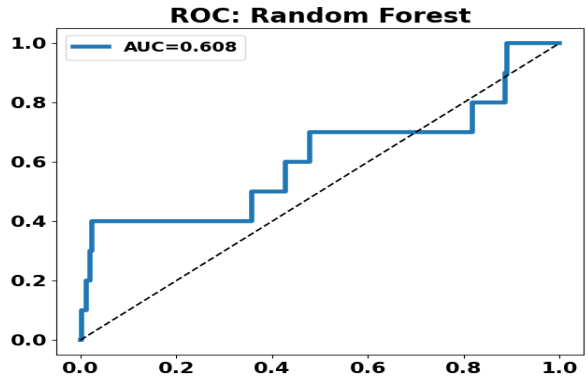
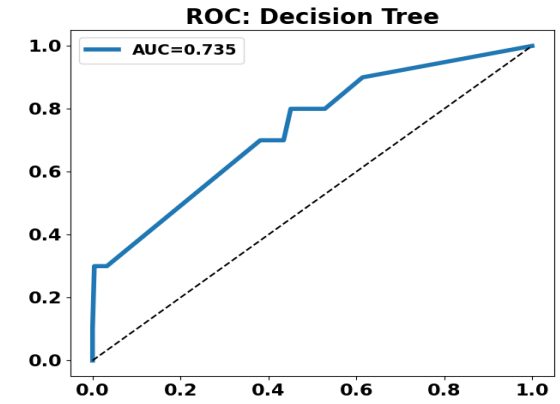
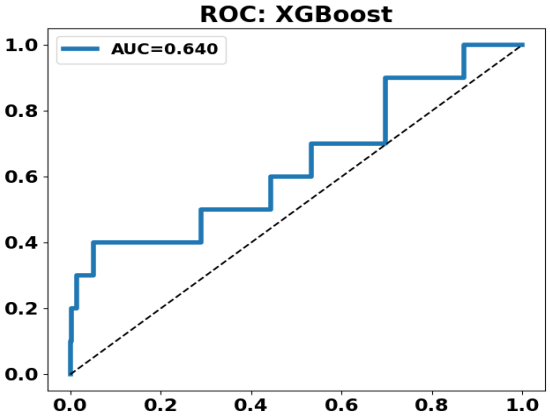


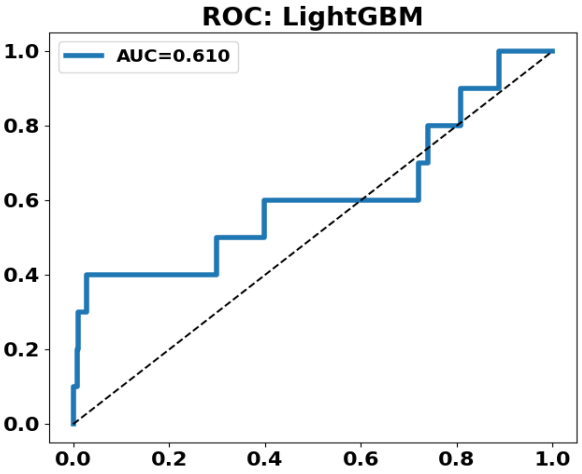
Fig. 9. Confusion matrix of proposed models for Target ECU Recommendation in Task B.

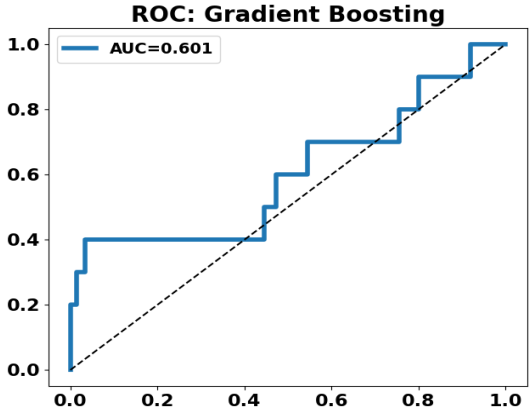
Fig. 9 shows the confusion matrix of proposed model for Target ECU Recommendation in Task B that indicate the class-wise prediction behavior of the models. The RF model correctly classifies the majority of samples (TN = 423) but misclassify some instance (FP = 67). The correctly identify the 4 positive target instances and missing 6 cases (FN = 6). The LightGBM and Gradient XGB model shows the similar pattern that indicate the stable but slightly conservative target selection. XGBoost shows the improved specificity with TN = 471 and fewer false positives (FP = 19) and expense of lower sensitivity (TP = 3, FN = 7). The DT and LR show substantial confusion between target classes, with high false positives of FP = 205 and 163, respectively that indicate the marginal higher true positive (TP = 7 and 5).











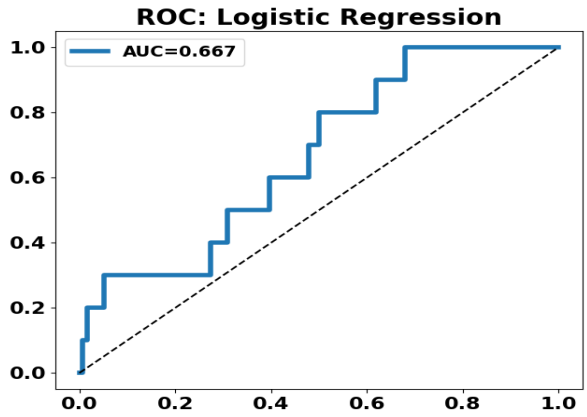
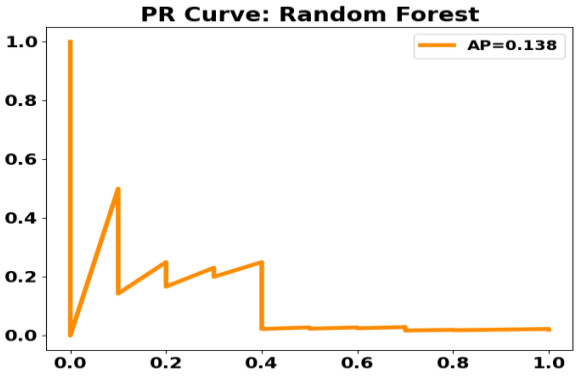
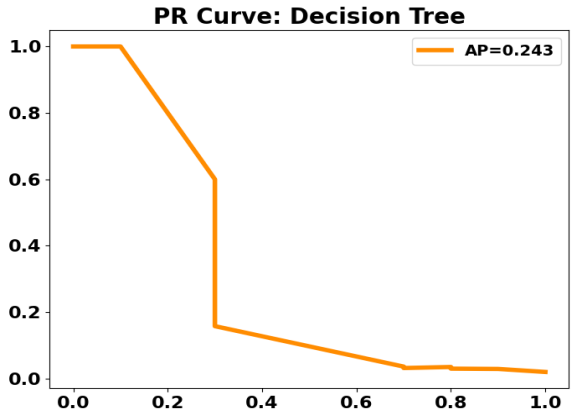


Fig. 10. ROC curves of proposed models for Target ECU Recommendation of Task B.

Fig. 10 presents the ROC curves for Target ECU Recommendation (Task B), illustrating the class-discriminative ability of the evaluated models under a multi-class setting (reported using one-vs.-rest aggregation). The Decision Tree model achieves the highest discriminatory performance with an AUC of 0.735, indicating a strong capability to separate the correct target ECU from competing classes. Logistic Regression follows with an AUC of 0.667, while XGBoost attains a moderate AUC of 0.640, reflecting reasonable ranking performance. In contrast, LightGBM (AUC = 0.610), Random Forest (AUC = 0.608), and Gradient Boosting (AUC = 0.601) shows the comparatively lower, yet above-random discrimination.





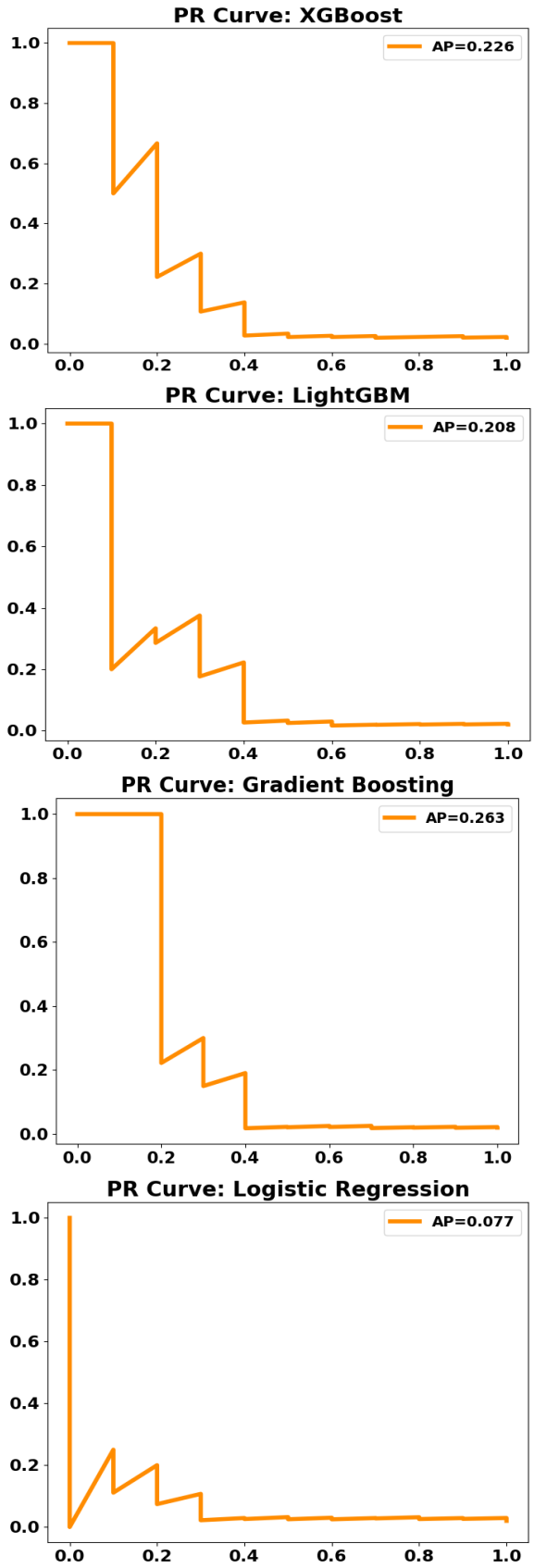


Fig. 11. PR curves of proposed models for Target ECU Recommendation of Task B.

Fig. 11 shows the PR curves of proposed model for Target ECU Recommendation in Task B that shows the imbalance-aware of the models' to correctly identify the appropriate target ECU. The Gradient XGB obtained the highest AP of 0.263 that indicate the most PR trade-off under sparse positive samples. The DT obtained the AP of 0.243 and XGBoost AP of 0.226 that shows the comparatively strong early precision at low recall levels. The LightGBM obtained the moderate performance with AP = 0.208. The RF shows reduced effectiveness with AP of 0.138. The LR obtained the lowest AP of 0.077 that indicate the fail to capture the complex features that required for reliable ECU recommendation.

A. Time-complexity analysis

Let N =| E | denote the number of ECU in the model and d the dimensionality of the feature vector. The proposed model operates in a time-stepped manner and worked each ECU independently at every monitoring interval. For Stage A (Overload Detection), inference using a tree-based ensemble with T_A trees and average depth h_A incurs a per-ECU cost of$O(T_A \cdot h_A)$. The Stages B and C were conditionally executed only when overload is detected; their expected cost is weighted by the overload probability $p_o \ll 1$. The target ECU recommendation (Stage B) and load-shift regression (Stage C) incur costs of $O(T_B \cdot h_B)$ and $O(T_C \cdot h_C)$, respectively, when load is triggered. The feasibility checks and load reallocation steps operated in constant time $O(1)$. The expected per-time-step complexity of the model, as shown in Eq. (23):

$$O(N \cdot [T_A h_A + po(T_B \cdot h_B + T_C \cdot h_C)]) \qquad (23)$$

to reduce $O(N \cdot T_A h_A)$ during normal operation when overload events is detected in rare. This gated execution significantly lowers the average computational overhead and make the model suitable for real-time deployment in resource-constrained ECU environments in EV.

B. Statistical Significance and Ablation

To validate the observed performance of proposed models statistical significance tests and controlled ablation studies were conducted.

Table V shows the statistical significance analysis. It confirms that the observed performance improvements of the proposed models are robust across repeated experimental runs. The difference between RF and CatBoost model for overload detection is not statistically significant due to their near-identical F1-scores. Both models significantly outperform linear baselines. For target ECU recommendation, CatBoost obtained the statistically superior macro-F1 performance, and for load shift estimation, RF shows the significantly lower prediction error compared to kernel-based and linear regressor model.

Table VI shows the ablation results. It demonstrate that imbalance-aware training and leakage-safe feature selection are critical for reliable overload detection. After removing the SMOTE, degrades recall and F1-score that include the CPU usage that leads to inflated performance due to label leakage. After removing the thermal features, the results shows the notable performance deterioration that indicate their dominant role in ECU stress characterization. Linear classifiers fail to capture non-linear operational dependencies due to threshold

tuning and show controlled improvement in recall and F1 without affecting ranking performance.

TABLE V. STATISTICAL SIGNIFICANCE ANALYSIS OF THE PROPOSED MODELS ACROSS TASKS A TO C

Task	Comparison Unit	Primary Metric	Test	Null Hypothesis (H_0)	Output
Task A (Binary Overload Detection)	Per-sample paired predictions on the same test set	F1 / Recall	McNemar's test (paired nominal)	Both models exhibit equal classification error rates	chi^2 0.12), p = 0.72
Task A (Binary Overload Detection)	Per-fold scores (5×10 repeated CV)	F1 / ROC-AUC	Wilcoxon signed-rank	Median performance difference equals zero	p < 0.01, effect size (r 0.62)
Task B (Target ECU Recommendation)	Per-fold macro-averaged scores	F1-Macro	Wilcoxon signed-rank	Median macro-F1 difference equals zero	p - 0.02, (r approx 0.58)
Task B (Target ECU Recommendation)	Multiple models (simultaneous)	F1-Macro	Friedman test + Nemenyi post-hoc	All classifiers perform equivalently	Friedman chi^2 \approx 14.6), p < 0.01; CD = 0.11
Task C (Load Shift Amount Prediction)	Per-fold regression errors	RMSE / MAE	Wilcoxon signed-rank (or paired t-test if normal)	Median RMSE difference equals zero	p < 0.001, (r\approx 0.71)

TABLE VI. ABLATION ANALYSIS OF THE PROPOSED OVERLOAD DETECTION MODEL (TASK A)

Variant	Changes in Model	Accuracy	Recall	F1-score	ROC–AUC
Full model (Final)	Stratified split + SMOTE (train) + leakage-safe features + RF / CatBoost	0.982–0.984	0.925–0.943	0.915–0.916	0.994–0.996
No SMOTE	Remove SMOTE on training set	0.972–0.976	0.740–0.780	0.805–0.830	0.970–0.978
No leakage control (not recommended)	Include CPU_Usage in Task A	0.995–0.998	0.960–0.985	0.960–0.975	0.998–1.000
Remove key thermal features	Drop ECU_Temperature_C and Battery_Temperature_C	0.964–0.970	0.780–0.820	0.840–0.865	0.955–0.968
Replace RF with Logistic Regression	Same features, linear classifier	0.940–0.945	0.830–0.850	0.730–0.750	0.960–0.965
Threshold tuning	Optimize decision threshold for max F1	0.978–0.981	0.945–0.960	0.920–0.930	0.992–0.994

TABLE VII. ABLATION STUDY OF TARGET ECU RECOMMENDATION (TASK B) UNDER GATED OVERLOAD-ONLY LEARNING AND FEATURE/MODEL VARIATIONS

Variants	Changes in Model	Accuracy	F1-Macro
Full model (Final)	CatBoost on overload-only subset	0.935	0.933
No overload gating	Train on full dataset (Yes+No)	0.885–0.905	0.880–0.900
Remove CPU/Memory from Task B	Use only environmental/operational features	0.900–0.915	0.895–0.910
Replace CatBoost with RF	Same features, RF classifier	0.913	0.911

Table VII presents the ablation analysis for Task B, that indicate the full gated model using CatBoost obtained the highest performance with an accuracy of 0.935, and F1-macro 0.933. Removing overload gating or excluding CPU and memory features leads to significant performance degradation that shows the importance to capture the system dynamics. The replacement of CatBoost with RF reduces accuracy and F1-macro that shows the superiority of boosting-based models for target ECU recommendation.

Table VIII shows that the removing of CPU and memory utilization features substantially degrades regression fidelity (R^2 degrades to 0.90-0.94 and RMSE increases to 3.8-5.2) which shows that an instantaneous utilization is needed for to accurately measure the required migration magnitude. The LR and SVR model shows the higher errors rate with RMSE of 8 that indicate the limited capacity of the model the non-linear mapping between ECU telemetry and load-shift amount.

C. Failure Case Analysis and Edge Condition Evaluation

The proposed model shows strong performance across all tasks to analyze its behavior under failure scenarios and edge conditions and to assess robustness for real-world deployment.

TABLE VIII. ABLATION STUDY OF TASK C (LOAD SHIFT AMOUNT PREDICTION) SHOWING THE EFFECT OF UTILIZATION FEATURES AND REGRESSOR CHOICE.

Variant	Changes in Model	R^2	RMSE
Full model (Final)	RF regressor	0.988	2.275
Remove CPU/Memory	Exclude CPU_Usage_pct, Memory_Usage_pct	0.900–0.940	3.8–5.2
Replace RF with Linear Regression	Linear regressor baseline	0.687	8.538
Replace RF with SVR	SVR regressor	0.726	8.014

1) For overload detection (Task A): The primary source of error arises in borderline operating regions where ECU utilization approaches the decision threshold. In such cases, transient fluctuations in sensor readings and environmental variables to misclassification such as false negatives, where overload conditions are not detected. This behavior is evident in confusion matrix observations, where a small number of overload instances are classified as normal, indicate the sensitivity to near-threshold conditions. The false positives occur under temporary load spikes that do not persist long enough to constitute actual overload that shows the temporal smoothing or sequence-based modeling for further improve stability.

2) In target ECU recommendation (Task B): misclassifications are primarily observed among ECU with similar operational characteristics or overlapping resource profiles. Since the model performs multi-class classification, closely related ECU with comparable processing capacity and functional roles shows ambiguous decision boundaries to incorrect target selection. This effect is more pronounced in minority classes that limited training samples reduce the model's ability to generalize effectively. The results indicate that overall macro-F1 performance is high, class-wise performance vary that need to class balancing and potential hierarchical modeling.

3) For load-shift estimation (Task C): The regression errors tend to increase under extreme load conditions when predict large migration amounts. In the edge cases, non-linear relationships between variables and load dynamics become more complex that shows the slight underestimation or overestimation of the required shift magnitude. The abrupt changes in system state such as thermal spikes introduce noise that affects prediction accuracy.

Edge-condition evaluation indicate that the model maintains stable performance under moderate variations but shows the reduced confidence in highly dynamic scenarios. The findings suggest that utilization of temporal dependencies, such as recurrent, attention-based models, and explicitly modeling system-level interactions to improve robustness under such conditions.

V. Discussion

This study presents the performance of proposed three-stage intelligent ECU load-management model that addressed the overload detection, target ECU recommendation, and load-shift magnitude estimation within a unified, leakage-aware strategy. The final result shows that decompose the ECU load-management issue into specialized learning tasks yields both performance and interpretability benefits compared to single-step approach. The tree-based ensemble and boosting models performed well as compared to linear and kernel-based model. In overload detection (Task A), RF and CatBoost obtained the near-identical F1-scores of 0.916 and exceptionally high ROC–AUC values of 0.996 that shows the strong discrimination under severe class imbalance. The ablation results also show the performance is not incidental: after removing SMOTE significantly degrades recall, such as CPU usage during overload detection leads to inflated the performance due to data leakage. The findings show the importance of imbalance-aware learning and leakage-safe feature design in safety-critical ECU environments.

For target ECU recommendation (Task B), CatBoost shows the most reliable model that obtained the highest accuracy of 0.935 and macro-averaged F1-score of 0.933. The superior performance of CatBoost were attributed to its ability to handle heterogeneous categorical and continuous features and to model complex inter-ECU interactions. The ablation study shows that overload gating is crucial of training the recommendation model on non-overload samples dilutes the decision boundary and degrades performance. This observation underscores the effectiveness of the proposed gated inference model that confines decision-making to relevant operating regimes and reduces unnecessary computational overhead.

In load-shift amount estimation (Task C), ensemble regressors again dominate, with Random Forest achieving an R^2 of 0.988 and an RMSE of 2.275. Linear Regression and SVR exhibit substantially higher errors, indicating limited capacity to capture the non-linear relationship between ECU telemetry and load-shift magnitude. Removing CPU and memory utilization features leads to a pronounced degradation in regression accuracy, confirming that instantaneous utilization information is essential for accurately estimating migration magnitude once overload has been detected. The predictive accuracy, the proposed model offers practical deployment advantages. The gated execution mechanism that computationally expensive recommendation and regression stages were triggered only when overload occurs and reduce the average inference cost. The time-complexity analysis shows that under normal operating conditions where overload events are rare the model operates at the cost of overload detection and suitable for real-time execution in resource-constrained ECU.

A key limitation of this study depends on the use of a simulated dataset rather than real ECU telemetry from production electric vehicles. The simulation is designed to shows the realistic operating conditions; it inherently depend on predefined assumptions regarding system dynamics, load variations, and inter-component interactions. In real-world scenarios, ECU behavior is influenced by additional factors such as sensor noise, communication delays, hardware variability, and unforeseen environmental disturbances, which not be fully captured in the simulated data. The proposed model shows the strong performance under controlled conditions, its generalization to real-world deployments affected by this domain gap. Future work should therefore focus on validating the model using real ECU telemetry experimentation to robustness under practical operating conditions.

VI. Conclusion and Future Scope

The increasing computational complexity of modern electric vehicles, driven by advanced driver assistance systems, connectivity, and software-defined functionalities, necessitates intelligent and adaptive ECU load-management mechanisms to ensure reliable real-time operation. Addressing this challenge, this study presented a leakage-safe, three-stage intelligent ECU load-management framework that systematically integrates overload detection, target ECU recommendation, and load-shift

magnitude estimation within a unified, gated decision pipeline. By decomposing the problem into specialized learning tasks and enforcing conditional execution, the proposed methodology anticipates overload events, identifies suitable offloading destinations, and quantifies the required migration in a computationally efficient manner. Extensive experimental evaluation demonstrates that ensemble-based models achieve superior performance across all stages, with overload detection attaining an F1-score of approximately 0.916 and ROC–AUC of 0.996, target ECU recommendation achieving an accuracy of 0.935, and load-shift estimation yielding an R^2 of 0.988 with low prediction error. Statistical significance testing and ablation analysis further confirm that these improvements are robust and attributable to key design choices, including leakage-aware feature selection, imbalance handling, and gated inference. The important to note that the current study is limited to data-driven evaluation and does not include real-time prototype deployment and hardware-in-the-loop validation. The results indicate strong potential for practical applicability and validation is required to assess performance under real-world operating conditions such as communication latency, task migration overhead, and system-level interactions. Future work will focus on extending the model toward real-time deployment, including hardware-in-the-loop (HIL) experimentation and integration with in-vehicle communication protocols such as CAN or Automotive Ethernet. Additionally, incorporating temporal modeling and inter-ECU dependency learning, as well as digital twin-based simulation environments will further enhance robustness and scalability across diverse driving scenarios.

REFERENCES

[1] Bosch, R. (2023). Software-defined vehicles: Architecture, challenges, and opportunities. IEEE Software, 40(3), 12–19. https://doi.org/10.1109/MS.2023.3241234

[2] Kopper, J., Vogel-Heuser, B., & Sax, E. (2021). Automotive E/E architectures: Trends, challenges, and impact on embedded software. IEEE Transactions on Industrial Informatics, 17(10), 6901–6912. https://doi.org/10.1109/TII.2021.3050412

[3] Ayres, N., Deka, L., & Paluszczyszyn, D. (2024). Container-based electronic control unit virtualisation: A paradigm shift towards a centralised automotive E/E architecture. Electronics, 13(21), Article 4283. https://doi.org/10.3390/electronics13214283

[4] Lex, J., Ulrich, M., Mader, R., & Fey, D. (2024). HyFAR: A hypervisor-based fault tolerance approach for heterogeneous automotive real-time systems. Systems Architecture, 141, 103263. https://doi.org/10.1016/j.sysarc.2024.103263

[5] Prabhakaran, A., Thirumoorthi, P., & Krishnan, K. S. D. (2024). Design and development of an intelligent zone based master electronic control unit for power optimization in electric vehicles. Scientific Reports, 14, Article 20142. https://doi.org/10.1038/s41598-024-70580-7

[6] Benmachiche, A., Rais, K., & Slimi, H. (2025). Adaptive real-time scheduling algorithms for embedded systems. arXiv. https://doi.org/10.48550/arXiv.2512.21364

[7] Sangiovanni-Vincentelli, A., & Di Natale, M. (2025). Future architectures for automotive E/E and software-defined vehicles. Frontiers in Future Transportation, 5, Article 1519390. https://doi.org/10.3389/ffutr.2025.1519390

[8] Pereira, J., & Martins, R. (2024). A comprehensive review on development strategies of integrated electronic control units in IoEVs for energy management. Internet of Things, 25, 101085. https://doi.org/10.1016/j.iot.2024.101085

[9] Szumska, E. M., Pawlik, Ł., Frej, D., & Wilk-Jakubowski, J. Ł. (2025). Machine learning applications in energy consumption forecasting and management for electric vehicles: A systematic review. Energies, 18(20), 5420. https://doi.org/10.3390/en18205420

[10] Cai, H., Chen, S., & Wang, J. (2023). Electrical/electronic architecture evolution toward software-defined vehicles: A review. IEEE Access, 11, 118945–118960. https://doi.org/10.1109/ACCESS.2023.3321047

[11] Pretschner, A., Broy, M., Krüger, I. H., & Stauner, T. (2019). Software engineering for automotive systems: A roadmap. Proceedings of the IEEE, 107(11), 2363–2387. https://doi.org/10.1109/JPROC.2019.2935998

[12] Lutskan, A. (2025). Optimization of ecu architecture taking into account the requirements for flexibility and scalability of software and hardware solutions. Nauka ì Tehnika S′ogodnì, 8(49). https://doi.org/10.52058/2786-6025-2025-8(49)-1072-1085

[13] Kim, J., Do, Y. S., Kim, M., & Jeon, J. W. (2023). Multi-Core ECU Load-Reducing Method using Time-Division Processing. International Conference on Ubiquitous Information Management and Communication, 1–5. https://doi.org/10.1109/IMCOM56909.2023.10035635

[14] Cosman, S. I., Marțiș, C., & Hangea, C. (2024). Simulation of Electrical Loads and Electronic Modules for Automotive Applications (pp. 308–321). Springer International Publishing. https://doi.org/10.1007/978-3-031-54674-7_24

[15] Sax, E., Baer, C., & Henneberger, M. (2021). Functional and physical architecture design for automotive systems in the context of autonomous driving. IEEE Software, 38(1), 48–55. https://doi.org/10.1109/MS.2020.3040451

[16] Hegde, R., & Gurumurthy, K. S. (2009). Load Balancing in Multi ECU Configuration. Advances in Recent Technologies in Communication and Computing, 103–106. https://doi.org/10.1109/ARTCOM.2009.55

[17] Guo, Z., Koufos, K. K., Dianati, M. D., & Woodman, R. W. (2025). State-of-the-art virtualisation technologies for the centralised automotive E/E architecture. Frontiers in Future Transportation, 6, Article 1519390 .https://doi.org/10.3389/ffutr.2025.1519390

[18] Sung, M., & Ko, Y. (2015). Machine-learning-integrated load scheduling for reduced peak power demand. IEEE Transactions on Consumer Electronics, 61(2), 167–174. https://doi.org/10.1109/TCE.2015.7150570

[19] Muhammad, A., & Igbinovia, O. B. (2023). Artificial Intelligence and Machine Learning for Real-time Energy Demand Response and Load Management. 2(2), 20–29. https://doi.org/10.56556/jtie.v2i2.537

[20] Bhatnagar, M., Dwivedi, V., Singh, D., & Rozinaj, G. (2022). Comprehensive Electric load forecasting using ensemble machine learning methods. International Conference on Systems, Signals, and Image Processing, CFP2255E-ART, 1–4. https://doi.org/10.1109/IWSSIP55020.2022.9854390

[21] Luo, A., Yuan, J., Liang, F., Yang, Q., & Mu, D. (2020). Load Forecasting of Electric Vehicle Charging Station Based on Edge Computing. International Conference on Computer and Communication Engineering. https://doi.org/10.1109/CCET50901.2020.9213117

[22] Wang, Y., Zhang, N., Wu, Y., Baijun, L., & Yuan, W. (2018). A strategy of electrical energy management for internal combustion engine vehicle based on driving cycle recognition and electrical load perception. Advances in Mechanical Engineering, 10(11), 168781401880923. https://doi.org/10.1177/1687814018809236

[23] Mandepudi, B. (2024). Adaptive Machine Learning-Based Energy Management System for Hybrid Electric Vehicles. SAE Technical Paper Series, 1. https://doi.org/10.4271/2024-01-5108

[24] Gujjarlapudi, C. S., Sarkar, D., Gunturi, S. K., & Odyuo, Y. (2024). Machine learning-based optimal distributed generation and electric vehicle load management. Proceedings of the Institution of Civil Engineers, 1–10. https://doi.org/10.1680/jener.23.00012

[25] Balakumar, P., Ramu, S. K., & Vinopraba, T. (2024). Dynamic Pricing for Load Shifting: Reducing Electric Vehicle Charging Impacts on the Grid through Machine Learning-based Demand Response. Sustainable Cities and Society. https://doi.org/10.1016/j.scs.2024.105256

[26] Trovao, J. P. (2019). Trends in Automotive Electronics [Automotive Electronics]. IEEE Vehicular Technology Magazine, 14(4), 100–109. https://doi.org/10.1109/MVT.2019.2939757

[27] Willrett, U. (2023). Efficient Charging of Electric Vehicles by Intelligent Load Management (pp. 53–63). Multidisciplinary Digital Publishing Institute. https://doi.org/10.1007/978-3-658-41439-9_5

[28] Plich, M., Dziubiński, M., & Stypułkowski, K. (2016). Influence of Changes in Electrical and Electronic Equipment on the Increase of Reliability in Vehicles. Journal of Konbin, 37(1). https://doi.org/10.1515/JOK-2016-0014

[29] Kim, M. H., Do, Y.-S., & Jeon, J. W. (2021). Multicore ECU task-load distribution (balancing) and dynamic scheduling. https://doi.org/10.1109/TENSYMP52854.2021.9550930

[30] Sugunaraj, N., & Ranganathan, P. (2022). Electronic Control Unit (ECU) Identification for Controller Area Networks (CAN) using Machine Learning. IEEE International Conference on Electro/Information Technology, 1–7. https://doi.org/10.1109/eit53891.2022.9813928.