Enhancing 5G and 6G Networks Through a Dynamic Dual-Stage Machine Learning Heuristic Framework for Selecting UEs as UE-VBSs

Iacovos Ioannou[†‡], S.V.Jansi Rani[*], Prabagarane Nagaradjane[§],
Christophoros Christophorou[‖‡], Vasos Vassiliou[†‡], Andreas Pitsillides[*¶]
[*]Department of CSE, Sri Sivasubramaniya Nadar College of Engineering, Chennai-India
[†]Department of Computer Science, University of Cyprus
[‡]CYENS Research Center
[§]Wireless Technology Lab, Department of ECE, Sri Sivasubramaniya Nadar College of Engineering, Chennai-India
[¶]Department of Electrical & Electronic Engineering Science, University of Johannesburg, South Africa (Visiting Professor)
[‖]University of Central Lancashire (UCLan) Cyprus, Larnaka, Cyprus

***Abstract*—Adapting mobile networks to the diverse and evolving demands of 5G and forthcoming 6G technologies requires flexible, efficient, and dynamic strategies—especially in ultra-dense environments and infrastructure-limited areas. This paper proposes a robust two-stage Machine Learning (ML) heuristic framework to dynamically select a group of User Equipment (UEs) to act as Virtual Base Stations (UE-VBSs) for network augmentation. In the first stage, Self-Organizing Maps (SOM) are employed to cluster UEs based on their spatial characteristics while preserving topological relationships, achieving a silhouette score of 0.64—a 30% improvement over conventional methods such as K-Means (0.46) and Mean-Shift (0.43). In the second stage, a Random Forest classifier enhanced via the Synthetic Minority Over-sampling Technique (SMOTE) attains an average accuracy of 97% and an F1-Score of 0.88 in identifying eligible devices to become UE-VBSs, outperforming recent frameworks that typically report accuracies ranging between 85% and 92%.**

Comparative evaluation results demonstrate that our two-stage ML heuristic framework not only improves clustering accuracy and UE-VBS classification but also consistently outperforms state-of-the-art clustering methods in terms of network sum rate, power consumption, and scalability. Specifically, across all device densities (i.e., 200, 400, 600, 800, and 1000 UEs), our approach achieves the highest sum rate—peaking at nearly 1.8 billion bps (or 1.8 Gbps) at 1000 UEs—thus surpassing methods such as Affinity Propagation and Grid-based Clustering. Furthermore, by intelligently selecting UE-VBSs, the framework significantly reduces power consumption by effectively minimizing redundant transmissions and interference, making it an energy-efficient solution for large-scale 5G networks. Although the complexity of SOM clustering and Random Forest classification introduces higher computational overhead, the resulting improvements in throughput, energy efficiency, and scalability justify this cost, making it a robust and practical solution for real-world deployments. Validated on both synthetic and real-world datasets, our findings underscore the efficacy, scalability, and high impact of employing robust unsupervised and ensemble learning techniques for dynamic network optimization in next-generation architectures, delivering up to a five-fold increase in network sum rate under high-density conditions compared to state-of-the-art approaches like grid-assisted clustering and affinity propagation.

***Index Terms*—Classification, Clustering, Small Cells, Two-stage Machine Learning Engine, Virtual Base Station, 5G, 6G**

I. Introduction

The deployment of 5G networks is still in its early stages, and ongoing research seeks to achieve ambitious goals such as lower energy consumption, higher data rates, seamless connectivity, increased bandwidth, minimal latency, and the capacity to support a larger number of users. Achieving these goals necessitates minimizing network congestion, improving traffic control, and ensuring better coverage, which requires the adoption of small cells (i.e., small base stations) under the macro base station (BS) umbrella [5], [38]. Consequently, widespread deployment of small base stations (SBSs) and the establishment of small cells that support various radio access technologies (RATs) are inevitable for realizing these objectives [4], [41]. However, large-scale deployment of small cells, especially in emerging ultra-dense heterogeneous networks (HetNets), presents significant challenges that must be addressed before their full benefits can be realized in 5G networks [15]. These challenges include:

1) **Capital Expenditure (CAPEX) Investments:** The substantial investments required by Mobile Network Operators (MNOs) to deploy a large number of SBSs. This includes costs associated with purchasing equipment, upgrading existing infrastructure, and integrating new technologies [46].
2) **Site Acquisition:** The need for numerous sites for BS equipment installation, alongside the considerable time and economic costs associated with their deployment. Securing suitable locations, negotiating with property owners, and adhering to local regulations can significantly delay deployment timelines [46].
3) **Energy Consumption:** The increased energy demands resulting from the dense deployment of SBSs lead to higher operational costs and environmental concerns.

Efficient energy management strategies are essential to mitigate these issues [25].

4) **Dynamic Mobile Traffic:** Existing small cell deployments often fail to efficiently address the dynamic nature of mobile traffic due to static positioning constraints, leading to asymmetric traffic distribution over time and space [25].
5) **Operational and Maintenance Costs:** The higher costs related to the operation and maintenance of a large number of BSs. This encompasses routine maintenance, troubleshooting, and the need for specialized personnel to manage the network infrastructure [46].

To overcome these challenges in ultra-dense 5G cellular networks [26], a User Equipment as Virtual Base Station (UE-VBS) approach has been proposed [11]. This approach enables the flexible deployment of virtual small cells within a mobile environment. By augmenting smartphones with base station and relay node functionalities, these devices can be transformed into versatile VBSs that complement the existing MNO infrastructure. Depending on network needs, user equipment (UE), enhanced with base station capabilities (henceforth referred to as UE-VBS), can serve as either a virtual small cell (VSC) to enhance coverage, capacity, and data rates in stressed hotspots with weak infrastructure, or as an intermediate virtual relay node (VRN) to facilitate efficient data flow within the radio access network [53]. A UE-VBS acting as a VSC or VRN can also share its link with other UE-VBSs or UEs in its vicinity, enabling direct communication without relying on a macro base station link [30].

The development of practical small-cell solutions that address these challenges is of growing interest in both the research community and industry. The latest 3GPP specifications [1], for instance, introduce self-organizing network (SON) solutions to reduce operating expenditures (OPEX) and CAPEX associated with the rapid increase in SBS numbers. To achieve a cost-effective and adaptable solution, we extend what was proposed in [11] and [32]. The earlier work introduced a novel virtual UE-VBS layer within the Evolved UMTS Terrestrial Radio Access Network (E-UTRAN) architecture that leverages the widespread distribution of mobile users (owners of eligible UEs) to create a dynamic network. MNOs can then dynamically control and deploy UEs as Virtual Small Cells (VSCs) or Virtual Relay Nodes (VRNs) based on demand. The mobility of users ensures that UE-VBSs align with dynamic location preferences, addressing the limitations of static small-cell deployments. This approach offloads CAPEX to mobile users who provide their smartphones as virtual base station equipment, eliminating the need for extensive site acquisition and expensive installations. Furthermore, it reduces operational and maintenance costs as users take responsibility for their own devices. This concept builds on the UE-VBS principles proposed in [11], [58] and focuses on addressing critical issues to maximize the potential of ultra-dense 5G networks. In this work, we propose and develop a two-stage machine learning (ML) framework to dynamically activate the most appropriate UEs (i.e., with high battery power, high memory capacity, high CPU processing power) to UE-VBSs at optimal locations (i.e., those near high data consumption areas), thereby maximizing the sum rate and ensuring cost-effective dense HetNet deployment. Our framework involves:

1) **Clustering UEs** based on their Euclidean distance from the BS. Through extensive experiments using competitive unsupervised ML techniques, including K-Means, Spectral clustering, Agglomerative clustering, BIRCH, Mean shift, Affinity propagation, OPTICS, and Self-Organizing Maps (SOM), we observed that the SOM algorithm performs best, particularly because of its ability to preserve topological relationships. SOM attained an impressive silhouette score of 0.64, outperforming other methods (such as ClusterGAN which achieved a score of 0.45) and conventional clustering approaches.
2) **Binary Classification** of UEs within each cluster to determine eligible devices to become UE-VBSs based on Quality of Service (QoS), Quality of Experience (QoE), data rate, distance from the BS, battery life, and processing power.

Our two-stage ML framework starts by using SOM-based clustering (without prior knowledge of the cluster count) to group UEs, and then applies a supervised classification phase, testing AdaBoost, Gradient Boosting, and Random Forest, to identify eligible UE-VBSs. Random Forest performed best (F1=0.86, Recall=0.80), and combined with SOM delivered 97% accuracy, confirming the approach's effectiveness. The comparative evaluation results demonstrate that our Two-Stage ML Heuristic framework not only improves clustering accuracy and UE-VBSs classification but consistently outperforms state-of-the-art clustering methods in terms of sum rate, power consumption, and scalability. Specifically, across all device densities (i.e., 200, 400, 600, 800, 1000), our approach achieved the highest sum rate, with a peak of nearly 1.8 billion bps at 1000 UEs, outperforming Affinity Propagation and Grid-based Clustering. Additionally, by optimizing UE-VBSs selection, our framework significantly reduced power consumption by effectively minimizing redundant transmissions and interference, making it a more energy-efficient solution for large-scale 5G networks. While our approach required a higher computational cost due to the complexity of SOM clustering and Random Forest classification, the resulting performance benefits in network throughput, energy efficiency, and scalability justify this additional processing overhead, making it a robust and practical solution for real-world deployments. In the final step of the algorithm to select the final UE-VBS (cluster head), we employ a heuristic algorithm accounting for data rate thresholds and distance constraints, ensuring efficient traffic aggregation and high QoS. Unlike GAPC and MAPC, our solution conducts a detailed comparative study (including deep learning methods like ClusterGAN) and uses interpretability tools (SHAP, LIME, QLattice, Eli5) to validate SOM + Random Forest. In addition, we integrate threshold- and distance-based constraints in a final heuristic step to ensure

efficient traffic aggregation and high QoS for each cluster. We also analyze computational complexity and scalability for up to 1000 UEs, discussing potential optimizations and parallelization to meet ultra-dense 5G/6G requirements.

The novelty of this approach lies in addressing the challenges of evolving mobile networks by leveraging the UE-VBS concept [11]. This study demonstrates the efficient integration of UE-VBS into 5G and beyond networks, significantly improving network performance and scalability. The primary objectives are to enhance data throughput, service quality, and network scalability in ultra-dense 5G environments through the dynamic deployment of UE-VBSs. Secondary goals include optimizing power consumption, reducing operational costs, and minimizing the need for extensive physical infrastructure deployment. These objectives are crucial for the successful deployment and operation of next-generation networks.

The main contributions of this paper are:

- A novel two-stage ML pipeline for UE-VBS selection, combining unsupervised SOM clustering with Random Forest classification.
- A thorough comparison with prior approaches (GAPC, MAPC), providing *novel* evidence on how topological-preserving clustering (SOM) outperforms competing methods in silhouette score and sum-rate improvements.
- A new heuristic selection algorithm that ensures scalable, flexible UE-VBS activation based on battery discharge rate, minimum data rate, and maximum allowable distance.
- Extended experiments with both synthetic and real-world datasets (Telecom Italia, OPNET) to validate performance gains in sum rate and energy considerations.
- **Unique Approach:** This paper is the first to employ SOM-based clustering for UE-VBS selection and the only work that classifies UEs based on specific criteria—including CPU, power, memory, and link quality—to qualify them as UE-VBS.

Following this introduction, the remainder of the paper is organized as follows. Section II provides background information and analyzes related research. Section III presents the problem, system, and scenario description. Section IV details the methodology, including a thorough description of the approaches and techniques used in this study. It covers the processes of selecting and comparing different classification and clustering methods, generating datasets, and selecting features. The section is structured to reflect the two main stages of the machine learning pipeline: clustering and classification, each with its specific methodologies and objectives. Section V presents the evaluation of the methodology, including the results of clustering, classification, and the heuristic algorithm for selecting UE-VBSs that form the mobile communications network. Section VI concludes the paper and outlines future directions.

II. Background And Related Work

In this section, we provide background information and related work on 5G/6G networks, especially in the context of user equipment such as Virtual Base Station (UE-VBS) selection and deployment, small cell selection, and ML-based approaches to create ultradense networks. This subsection presents the background information regarding UE-VBS selection, deployment, and ML. In addition, this section provides the related work of clustering and classification approaches in telecommunications, thus the work related to our investigation. Following the related work, we have a brief description and a comparison of the techniques used in our work, contrasting with some papers in the related work. Finally, it provides the identified research gaps that, through our research, we want some of them to resolve.

A. Background Work

The deployment of Small Base Stations (SBS) such as femtocells, picocells, and nanocells is typically static and driven by demand, particularly around hotspots. However, this static deployment can be inefficient in scenarios with unpredictable traffic surges. To address this, a UE-VBS solution is proposed [11] for managing mobile traffic dynamics in ultra-dense 5G networks [35]. This solution facilitates the flexible real-time deployment of numerous Virtual Small Cells (VSC).

The Affinity Propagation Clustering (APC) method, discussed in [53], is employed to select the most suitable UEs from a pool of UE-VBSs to support an overloaded SBS. This approach combines mobile networks and ML, dynamically identifying a subset of UEs served by the macro BS that can act as UE-VBSs. APC addresses the challenge of device classification, enabling a device to function as a small cell UE-VBS, a topic not extensively covered in the existing literature. The algorithm groups similar objects by considering all data points as potential cluster centers, selecting the UE-VBS for each cluster based on Euclidean distance and transmission range (Min and Max). An Initializing Matching Connection Algorithm (IMCA) is proposed in [58]. IMCA focuses on establishing initial connections between client UEs and eligible UEs (future VSCs) based on the Signal-to-Interference plus Noise Ratio (SINR) preference, forming clusters. The algorithm then optimizes these clusters within upper and lower bound constraints and cluster size, creating a UE-VBS.

The functioning aspects of UEs as UE-VBSs are further explored in [48], [58], and [53]. In [48], a scenario is considered in which the BS has limited capacity, service connections, bandwidth, and power and cannot meet the demands of several UE clients. To mitigate this, the authors suggest that a BS can select a capable subset of UEs to serve nearby UEs by becoming UE-VBSs. This research assumes that the selected UEs have the necessary hardware and software specifications. The selection of a UE to become an eligible UE-VBS is based on the UE distribution and the number of UEs associating with the active UE-VBS based on the maximum received power.

In this paper, we leverage a broad range of machine-learning algorithms for both clustering and classification. For clustering, we investigate K-Means, Mean-Shift, Affinity Propagation, SOM, Spectral Clustering, Agglomerative Clustering, BIRCH, OPTICS, and a deep-learning–based ClusterGAN.

For classification, we explore Random Forest, AdaBoost, Gradient Boosting, k-Nearest Neighbors (KNN), Naive Bayes (NB), Support Vector Machines (SVM), and Linear Discriminant Analysis (LDA). A concise overview of these methods, along with their key parameters, computational complexity, and main characteristics, is presented in Table I. For more in-depth discussions on how these algorithms operate and why they are suited for our UE-VBS selection framework, please see Appendix A, Section A.

B. Related Work

In our study, we utilize clustering (unsupervised) and classification (supervised) techniques to develop a machine-learning model for selecting UEs to act as UE-VBSs, thereby dynamically forming the network using Virtual Small Cells (VSCs). In this section, we present the related work that aligns with and is most relevant to the research problem investigated in this paper.

The authors in [13] introduce the Grid Assisted Affinity Propagation Clustering (GAPC) algorithm, which advances the concept of Cell Head based Virtual Small Cell (CHVSC) service. By optimizing the selection of Cell Heads (CHs) through grid zonal division and a Depth-First Search algorithm, GAPC significantly improves total spectral efficiency. GAPC achieves a silhouette score of 0.9640, compared to MAPC, and appoints more CHs, enhancing bandwidth efficiency and reducing computational complexity. However, MAPC records a higher average Signal-to-Interference-plus-Noise Ratio (SINR), suggesting a trade-off between SINR and clustering performance.

The research in [59] explores UE-based Virtual Base Station (called UE-VBS) technology, utilizing Modified Affinity Propagation Clustering (MAPC) and Multi-hop Load Balanced Geographical Path selection (MLGP) algorithms. These algorithms dynamically create virtual small cells (VSCs) and virtual relay nodes (VRNs) to expand network capacity in areas with weak infrastructure. MATLAB and NS3 simulations demonstrate the effectiveness of these approaches through improved network throughput (59,167 kbps for 1000 UEs), reduced delay (501 ms for 1000 UEs), and minimized jitter (94 ms for 1000 UEs), highlighting the algorithms' capabilities in enhancing 5G network robustness and efficiency.

In [60], an innovative framework is proposed to optimize power consumption within 5G heterogeneous networks by integrating an Intelligent Backhauling Algorithm (IBA) and Load Sharing Based s-BS Sleeping (LSBS) strategy. While specific numerical values are not provided, the framework is claimed to outperform existing algorithms in balancing energy efficiency and network performance.

The authors in [24] examine incorporating AI into autonomous wireless systems design. Unlike data-driven approaches, this work motivates and outlines a general AI methodology for autonomous agents in real-time self-organization, unifying knowledge management with sensing, reasoning, and active learning. They extend machine intelligence concepts to wireless systems, highlighting training-free and training-based methods, with reinforcement learning as a key example, and provide a functional example of an autonomous agent.

In [19], two power control (PC) algorithms based on distributed Q-learning and a decision tree classifier (CART) are proposed for D2D communication in underlay cellular networks. Compared with traditional PC methods, these algorithms optimize computational complexity, communication throughput, and energy efficiency. They address PC for single D2D pairs and proportional fair scheduling for multiple D2D pairs and cellular users, improving system fairness and efficiency.

The innovative work in [31] proposes a distributed AI framework using Belief-Desire-Intention eXtended (BDIx) agents and a transmission mode selection plan called "DAIS" for a D2D communication network. This framework aims to maximize data rates and minimize power consumption while accounting for computational load. Simulation results demonstrate the utility of BDIx agents in solving D2D communication issues and suggest that distributed AI is optimal for link realization.

Contributions in [34] explore strategies for improving spectral efficiency by invoking small cells, coordinated multipoint, and massive MIMO. The authors adopt a successive UE selection strategy, demonstrating a 100% improvement in spectral efficiency and suggesting a potential 1000-fold increase in mobile traffic with a smart combination of the presented techniques.

The work in [42] analyzes the performance of small cells using unmanned aerial vehicles (UAVs), addressing challenges such as deployment, power consumption, coverage optimization, and interference management. They derive expressions for ground-to-air channel models and show that optimal drone small cell (DSC) separation distance maximizes coverage, providing insights for more general DSC deployment cases.

A multi-variant clustering approach with interference alignment to improve spectral efficiency in densely packed small cells is introduced in [51]. An experiment with a network of 50 small cells demonstrates that the proposed approach suppresses path loss and interference, resulting in improved spectral utilization and maximum bits transfer.

In [7], an energy-efficient clustering approach for wireless sensor networks (WSNs) using a Type-3 fuzzy system is presented. This algorithm addresses uncertainties in WSNs, such as node residual energy, cluster centrality, and distance from the cluster head, optimizing energy consumption while maintaining network performance. The use of Type-3 fuzzy logic effectively handles imprecision in node localization and energy estimation, leading to robust clustering decisions.

Lastly, in [61], the authors address cross-domain recommendations to enhance recommender systems by leveraging knowledge from multiple domains. They propose a Generalized Matrix Factorization (GMF) model integrating latent factors from different domains, which improves recommendation accuracy. Comprehensive experimental results validate the model's effectiveness in various cross-domain scenarios.

TABLE I: Summary of Clustering and Classification Algorithms Used in Our Approach

Algorithm	Type & Key Parameters	Complexity	Comment
	Clustering Methods		
K-Means	*Clustering* Number of clusters k, initialization method	$O(n\,k\,t)$	Simple, fast for moderate n; requires specifying k.
Mean-Shift	*Clustering* Bandwidth (kernel size)	$O(n^2\,t)$	Finds modes automatically; no k needed; can be slow for large n.
Affinity Propagation	*Clustering* Damping factor, preference	$O(n^2\,t)$	No k needed; can be memory-intensive for large n.
SOM	*Clustering* Grid size, learning rate	$O(n^2)$	Preserves topology; well-suited for high-dimensional or non-linear data.
Spectral Clustering	*Clustering* Number of clusters k, affinity matrix	$O(n^3)$	Uses graph Laplacian; handles complex boundaries but less scalable for large n.
Agglomerative Clustering	*Clustering* Number of clusters k, linkage criterion	$O(n^2)$	Hierarchical method; can produce a dendrogram of clusters.
BIRCH	*Clustering* Number of clusters k, branching factor	$O(n)$	Excellent for large datasets; incremental and memory-efficient.
OPTICS	*Clustering* MinPts, neighborhood radius (ϵ)	$O(n^2)$	Density-based; discovers clusters of arbitrary shape, but can be slow at scale.
ClusterGAN	*Clustering* GAN components: generator, discriminator, latent dimension	$O(n^2\,t)$	Deep-learning approach; learns clusters in latent space; can be computationally heavy.
	Classification Methods		
Random Forest	*Classification* Number of trees T, maximum depth	$O(T \cdot n \log n)$	Robust ensemble; less prone to overfitting; parallelizable.
AdaBoost	*Classification* Number of estimators T, learning rate	$O(T \cdot n)$	Sequential boosting approach that adapts to errors; can struggle with noisy data.
Gradient Boosting	*Classification* Number of estimators T, learning rate, max depth	$O(T \cdot n)$	Iteratively fits new models to correct errors in previous ensemble steps.
k-Nearest Neighbors (KNN)	*Classification* Number of neighbors K, distance metric Simple method; can be slow at prediction time for large n; no training phase.	$O(n\,m)$	
Naive Bayes (NB)	*Classification* Prior & conditional probabilities	$O(n\,m)$	Very fast; assumes feature independence; suitable for high-dimensional inputs.
Support Vector Machines (SVM)	*Classification* Kernel type, C, γ	$O(n^2)$ to $O(n^3)$	Effective in high-dimensional spaces; can be slow or memory-heavy for large n.
Linear Discriminant Analysis (LDA)	*Classification* Covariance matrix, class priors	$O(n\,m^2)$	Projects data onto lower-dimensional subspace; requires normally distributed features.

Recent advancements in UE-VBS and small cell networks have significantly emphasized improving energy efficiency and network performance through advanced clustering and machine learning techniques. For example, Ioannou et al. [32] present a machine learning-aided approach for dynamic clustering and classification of UEs as UE-VBS in device-to-device (D2D) communication networks. This study utilizes K-Means combined with Mean Shift clustering to accurately group UEs into clusters and employs various classification algorithms, including Decision Trees, to predict the eligibility of UEs to serve as VBSs. The approach results in optimized load distribution, improved data rates, and enhanced network performance, demonstrating a significant increase in network efficiency, particularly in high-density environments.

Similarly, Danisya et al. [14] introduce the Grid Assisted Affinity Propagation Clustering (GAPC) algorithm for mmWave D2D in virtual small cells. This method builds upon previous clustering techniques by incorporating grid zonal division and a Depth-First Search algorithm for advanced eligible-UE selection. The GAPC algorithm enhances spectral efficiency, optimizes cluster head selection, and reduces computational complexity, leading to more efficient resource utilization and alleviation of traffic burdens in ultra-dense network scenarios.

Furthermore, Vaishnav et al. [56] propose an improved Affinity Propagation (AP) clustering algorithm specifically tailored for D2D communication in 5G networks. The enhanced AP algorithm incorporates weightage criteria and historical cluster information to refine the similarity index calculations, resulting in more stable clusters with appropriate cluster heads. Simulation results indicate that the proposed method significantly reduces the number of clusters and cluster heads, enhancing overall network efficiency. This research contributes to the optimization of D2D communication, a key technology for realizing the full potential of 5G networks.

These recent contributions underscore the ongoing efforts to enhance the efficiency and effectiveness of UE-VBS systems, particularly in the context of emerging 5G and 6G networks. The techniques proposed in these studies are complementary to the approach we adopt in this paper, where we further refine

the clustering and classification processes to optimize network performance. Our investigation reveals that the proposed classification model achieves 97% accuracy in determining an UE's eligibility to serve as a UE-VBS. The selected UE-VBS demonstrates significantly improved network metrics, including a nearly 500% data rate improvement, surpassing the data rates reported in other studies. We also conduct an in-depth analysis of various clustering algorithms to optimize clustering and explain the rationale behind our ML model, quantified by the results obtained.

C. Comparative Study among the Investigated Approaches and Research Gaps

This section compares various approaches that enhance the efficacy and efficiency of 5G and beyond networks using UE-VBS or D2D-Relays, focusing on methodologies involving ML, selection for power-efficient devices, two-stage approaches, and evaluation of device capabilities with an emphasis on QoS/QoE and accuracy.

Table II highlights the performance of various clustering algorithms. The Grid Assisted Affinity Propagation Clustering (GAPC) algorithm achieves the highest clustering accuracy with a silhouette score of 0.9640 [14], surpassing all other methods in the Accuracy column. Additionally, our two-tiered ML framework for UE-VBSs selection attains an impressive classification accuracy of approximately 97% using Random Forest and SOM algorithms, leading in both the QoS and QoE columns. While algorithms like MAPC and MLGP excel in throughput and delay [59], while others focus on power optimization [60] or spectral efficiency [34], our approach provides superior overall performance by balancing high accuracy with enhanced QoS and QoE. These results demonstrate the effectiveness of our methodology in optimizing network performance across multiple key metrics.

However, despite significant advancements in optimizing network performance through various clustering and classification approaches in 5G and beyond networks, several research gaps persist that need to be addressed for further improvement:

1) **Integration of Multi-Stage Machine Learning Pipelines**: Existing studies often focus on single-stage clustering or classification techniques. There is a lack of comprehensive approaches that integrate both stages into a unified pipeline. Our approach addresses this gap by implementing a robust two-stage machine learning pipeline to optimize UE-VBS selection and enhance network metrics such as data rate, QoS, and QoE.
2) **Evaluate Device Capabilities Before Use as UE-VBS**: Many methodologies do not thoroughly evaluate devices' capabilities for specific roles like UE-VBS. Our approach emphasizes evaluating device capabilities using a detailed suite of machine learning techniques, ensuring the selection of only the most suitable devices to enhance overall network performance.
3) **Optimization of Power Consumption in Dynamic Networks**: While frameworks like the Intelligent Backhauling Algorithm (IBA) and Load Sharing-Based BS Sleeping (LSBS) strategy optimize power consumption, there is a need to integrate these strategies with dynamic network formations such as VSCs and UE-VBSs. Our work lays the foundation for such integration.
4) **Comprehensive Performance Metrics**: Many studies improve specific metrics like spectral efficiency or throughput but lack a holistic evaluation encompassing computational complexity, energy efficiency, QoS, and QoE. Our two-stage pipeline addresses this by delivering significant improvements across a wide range of performance metrics.
5) **Real-World Implementation and Scalability**: Most research relies heavily on simulations. There is a need for real-world implementations to validate the scalability and practical applicability of these algorithms and frameworks. Future research should transition from simulated environments to real-world scenarios to test and refine these methodologies.
6) **Exploration of Advanced Clustering Techniques**: While methods like GAPC and MAPC show promise, further exploration of advanced clustering techniques and their combinations is necessary. Our in-depth analysis of various clustering algorithms, including Mean-Shift, Affinity Propagation, OPTICS, K-Means, Spectral Clustering, Agglomerative Clustering, and BIRCH, provides a pathway for future research to build upon and optimize these techniques.
7) **Addressing the Trade-Offs Between Metrics**: Balancing different performance metrics, such as SINR and clustering performance, remains a common challenge. More research is needed to develop algorithms that can effectively balance these trade-offs. Our findings highlight the need for nuanced solutions that do not compromise one metric for another.
8) **Advanced AI and Reinforcement Learning Techniques**: There is potential for further integration of advanced AI and reinforcement learning techniques into network optimization frameworks. Approaches like BDIx agents for D2D communication present opportunities to enhance decision-making processes in real-time network scenarios.

By addressing these research gaps, future research can significantly advance the capabilities and performance of 5G and beyond networks, paving the way for more efficient, reliable, and high-performing network infrastructures. Our approach specifically focuses on resolving the **Integration of Multi-Stage Machine Learning Pipelines**, **Evaluating Device Capabilities Before Use as UE-VBS**, and **Real-World Implementation and Scalability**, thereby contributing to the overall enhancement of QoS and QoE in modern network architectures.

III. Problem, System and Scenario Description

A. Problem Description

The rapid development of advanced mobile devices, including smartphones, tablets, and wearables, which support voice,

TABLE II: Comparative Analysis of Approaches Enhancing Efficacy and Efficiency in 5G and Beyond Networks and Gaps Identification

Reference	Approach	ML Used	Selects Powerful Devices	Two-Stage Approach	Evaluating Device Capabilities	QoS	QoE	Accuracy
[32]	ML-aided clustering and classification of UEs as VBS	✓	✗	✓	K-Means with Mean Shift clustering and Decision Trees for VBS eligibility	✓	✓	Optimized throughput and reduced energy consumption
[14]	GAPC for mmWave D2D in virtual small cells	✓	✗	✗	Grid zonal division and Depth-First Search algorithm	✓	✓	High (0.9640 silhouette score)
[13]	GAPC algorithm for CH-based VSC service	✗	✗	✗	Grid zonal division, DFS for CH selection	✓	✓	Not Specified
[59]	UE-VBS deployment with MAPC and MLGP	✓	✗	✓	Dynamic creation of VSCs and VRNs	✓	✓	Not Specified
[60]	Power consumption optimization	✗	✗	✗	IBA and LSBS strategy	✗	✗	Not Specified
[24]	AI in autonomous wireless systems	✓	✗	✗	Knowledge management with sensing, reasoning, and active learning	✓	✗	AI-driven
[19]	PC algorithms with ML for D2D	✓	✗	✗	Q-learning and CART	✓	✗	Better than traditional PC methods
[31]	Distributed AI for D2D using BDIx agents	✓	✗	✗	ML and reinforcement learning	✓	✓	Demonstrated
[34]	Spectral efficiency improvement strategies	✗	✗	✗	Successive selection of UEs	✓	✗	Effective
[42]	UAVs for small cells	✗	✗	✗	Ground-to-air channel models	✓	✓	Established
[51]	Clustering with interference alignment	✓	✗	✗	Multi-variant clustering approach	✓	✗	Significant
[7]	Grid-based clustering using Type-3 fuzzy system	✓	✗	✗	Handles uncertainties in node localization and energy estimation	✓	✗	Robust and efficient clustering decisions
[61]	Cross-domain recommendation using GMF	✓	✗	✗	Integrates latent factors from different domains	✓	✗	Significant improvements over traditional methods
[56]	Improved AP clustering for D2D in 5G	✗	✓	✗	Dynamic clustering with enhanced AP algorithm	✓	✓	Significant reduction in clusters and cluster heads
Our Approach	Two-tiered ML framework for UE-VBS	✓	✓	✓	Clustering: Mean-Shift, Affinity Propagation, OPTICS, K-Means, Spectral Clustering, Agglomerative Clustering, BIRCH, SOM; Classification: Random Forest, AdaBoost, Gradient Boosting	✓	✓	Approximately 97% accuracy with Random Forest and SOM for classification

data roaming, and video streaming, has led to an increased demand for fast network access and high-quality connectivity. These advancements require more bandwidth than traditional high-tower base stations can provide. To meet these demands, the 5G cellular network introduces stringent requirements, necessitating changes across all network layers and the use of small cells for ultra-densification [5].

Ultra-densification in 5G builds on the concept of heterogeneous networks (HetNets), first introduced in 4G LTE. This approach involves adding layers of smaller cells, such as picocells, femtocells, and dispersed antenna elements, to increase capacity, manage the growing number of user terminals, and alleviate congestion at 4G base stations [48]. These small cells enhance energy efficiency and reduce infrastructure costs while increasing network capacity. Strategic placement of microcells and relays within the macrocell range can offload the macrocell, conserve energy, and improve coverage, particularly in high-density environments like offices, malls, and public spaces. The bandwidth requirements for various applications vary significantly, with standard-definition video streaming needing 3-4 Mbps and high-definition video streaming requiring 5-8 Mbps. Online gaming demands around 3 Mbps with low latency, while video conferencing needs stable bandwidth between 1-4 Mbps to avoid disruptions. Advanced applications such as augmented reality (AR), virtual reality (VR), and holographic communications have even higher requirements, with AR/VR often exceeding 20 Mbps and holographic communications requiring up to 1 Tbps with latency under 1 ms. Applications like remote surgery and autonomous vehicles also demand extremely low latency and high reliability, with bandwidths ranging from 1 Gbps to 10 Gbps.

Meeting these requirements poses several challenges:

1) **Spectrum Availability**: The limited radio frequency spectrum makes it challenging to allocate sufficient bandwidth for high-demand applications.
2) **Increased Number of Small Cells**: Deploying numerous small cells for ultra-densification increases network management complexity and operational costs.
3) **5G Network Complexity**: The architecture of 5G networks, with multiple cell layers and heterogeneous integration, adds significant complexity to network management and seamless connectivity.
4) **Energy Consumption**: The dense deployment of small cells and complex network operations result in higher energy consumption, necessitating energy-efficient solutions.

Research focuses on optimizing network architectures and developing energy-efficient technologies, such as advanced beamforming, dynamic spectrum sharing, and intelligent network management. Additionally, using renewable energy sources for cell sites is being explored to reduce operational costs and environmental impact. The evolution towards 6G networks aims to overcome these challenges by supporting data rates up to 1 Tbps, latency as low as 1 ms, and extensive device connectivity per square kilometer, thus meeting the growing demands of advanced applications.

Therefore, this paper proposes a two-stage ML framework to activate UE-VBSs dynamically near areas of high data consumption, enhancing service quality across clusters. The solution involves two stages: (i) clustering UEs based on their distance from the BS, and (ii) binary classification of UEs in each cluster into eligible and ineligible UEs, considering factors such as device QoS, QoE, data rate, distance from

the BS, battery power, and processing power. A heuristic algorithm then selects a UE as the UE-VBS (cluster head) based on classification results, focusing on data rate and Channel Quality Indicator (CQI) from surrounding devices. The proposed model aims to dynamically identify a UE-VBS node to aggregate mobile data traffic, ensuring QoS, maintaining high data rates, and leveraging the superior capabilities of modern UEs. The approach involves implementing a network composed of Virtual Small Cells (VSCs) by clustering UEs and selecting eligible UEs to act as UE-VBSs and cluster heads. The final goal is to enhance mobile broadband (eMBB) and massive machine-type communications (mMTC) in a network with a larger number of devices than those provided by the operator, thus fulfilling some of the critical 5G requirements [2].

B. System Description

The proposed system operates within a heterogeneous network environment, consisting of a macro base station (MBS) and multiple small cells (SCs), which include femtocells, picocells, and UE-VBS. The network serves a set of N UEs distributed across the coverage area.

Each UE i is characterized by its spatial coordinates (x_i, y_i), residual battery power P_i, and computational resources. UEs are dynamically selected to function as UE-VBSs based on their proximity to areas with high data demand, sufficient battery power, and capacity to manage traffic from neighboring UEs. The network is segmented into clusters, where each cluster k is anchored by a UE-VBS that operates as a local base station for other UEs in the cluster.

1) Objective Functions: The system's primary goals are to maximize the network's sum rate and minimize its power consumption. These objectives are mathematically defined as follows:

1) **Maximization of Sum Rate:** The sum rate R_{sum} is the total data rate achieved across all UEs served by the UE-VBSs in the network. It is expressed as:

$$R_{\text{sum}} = \sum_{k=1}^{K} \sum_{i \in \mathcal{C}_k} B \log_2 \left(1 + \text{SINR}_{i,k}\right), \tag{1}$$

where $\mathcal{C}_k$ represents the set of UEs served by the k-th UE-VBS, B is the channel bandwidth, and $\text{SINR}_{i,k}$ denotes the Signal-to-Interference-plus-Noise Ratio of UE i when connected to UE-VBS k.

2) **Minimization of Power Consumption:** The total power consumption P_{total} of the network includes both the power used by the UE-VBSs for transmission and the power consumed by UEs communicating with their respective UE-VBSs. It is formulated as:

$$P_{\text{total}} = \sum_{k=1}^{K} \left(P_{\text{VBS},k} + \sum_{i \in \mathcal{C}_k} P_{i,k} \right), \tag{2}$$

where $P_{\text{VBS},k}$ is the power consumed by the k-th UE-VBS, and $P_{i,k}$ is the power consumed by UE i to communicate with UE-VBS k.

2) Optimization Problem: The system aims to achieve a balance between maximizing the sum rate and minimizing power consumption. The optimization problem can be stated as:

$$\max_{\mathcal{C}_k, \text{VBS}_k} \left(R_{\text{sum}} - \lambda P_{\text{total}} \right), \tag{3}$$

subject to the following constraints:

$$\sum_{k=1}^{K} P_{\text{VBS},k} \leq P_{\text{max}}, \tag{4}$$

$$\sum_{i \in \mathcal{C}_k} P_{i,k} \leq P_{\text{UE,max}}, \quad \forall k, \tag{5}$$

$$\text{VBS}_k \text{ is selected based on QoS, QoE, SINR, and residual battery.} \tag{6}$$

Here, λ is a weighting factor that manages the trade-off between the sum rate maximization and power consumption minimization. The constraints ensure that the power consumed by each UE-VBS does not exceed a maximum allowable limit P_{max}, and that the power consumption of each UE $P_{i,k}$ remains within the operational limits $P_{\text{UE,max}}$.

3) UE-VBS Selection Process: The selection of UE-VBSs follows a two-stage machine learning framework:

1) **Clustering Stage:** UEs are clustered based on their Euclidean distance from the macro base station using clustering algorithms like K-means. The number of clusters K is optimized to balance the load among UE-VBSs and to maximize the overall network sum rate.
2) **Classification Stage:** Within each cluster, UEs are classified into eligible and ineligible categories using a Random Forest classifier. The classification is based on metrics such as SINR, residual battery power, and data rate capabilities. The UE with the highest classification score within each cluster is selected as the UE-VBS.

The selected UE-VBSs serve as cluster heads, managing the aggregated traffic of their respective clusters while optimizing the network's sum rate and minimizing power consumption.

C. Scenario Description

In our work, the considered scenario is realized with the use of a two-stage ML engine, targeting the activation of the UE-VBSs at the "correct" locations at the time of need, thereby conceiving the dynamic deployment of VSC or VRN that can aid in extending data rates, capacity, and backhauling capabilities, thereupon resulting in enhanced service quality across the cell.

More specifically, in the first stage, we perform clustering using competitive and well-known unsupervised ML techniques such as K-Means (as shown in [3]), Spectral Clustering (as shown in [45]), Agglomerative Clustering (as shown in [49]), BIRCH (as shown in [62]), Mean Shift (as shown in [10]), Affinity Propagation (as shown in [21]), OPTICS (as shown in [6]), and SOM (as shown in [37]). In the second stage, binary classification is performed using competitive supervised ML techniques such as AdaBoost (as shown in

[54]), Gradient Boosting (as shown in [29]), and Random Forest (as shown in [39]).

Based on the results foregathered, the best-performing clustering and classification approaches are chosen to determine the eligible UEs that will be activated into UE-VBSs. In this work, we opted for a SOM Machine Learning approach with a silhouette coefficient of 0.64 and a Random Forest classifier with an F1-Score of 0.86 and Recall score of 0.80, as these models outperformed other considered ML models. Furthermore, to improve the model accuracy, the dataset used for training the Random Forest classifier is further improved by using the Synthetic Minority Oversampling Technique (SMOTE) approach in the unbalanced data set, aiming to overcome the inherent bias exhibited as a result of the majority[1] class (eligible UEs). Finally, the model is subjected to statistical testing and analysis using SHapley Additive exPlanations (SHAP) and Local Interpretable Model-agnostic Explanations (LIME). More details are provided in Section V-D4.

IV. Methodology

This section provides a comprehensive methodology that spans the entire data pipeline—from dataset generation and feature engineering to clustering, classification, and final UE-VBS selection. Multiple snapshots of UE distributions are taken at varying population densities to achieve a more realistic representation of real-world conditions. The parameters `memory`, `instantaneous_battery_percentage`, and `cpu_power` are randomly assigned, while additional parameters such as `battery_discharge_rate`, `distance_from_base_station`, `received_power`, and `signal interference noise ratio (SINR)` are computed using analytical expressions.

An overview of the entire methodology is depicted in Fig. 1. As shown, we first record UE positions, mobility patterns, and device capabilities to form a dynamic dataset. We then address key steps such as feature engineering, clustering (under various algorithms), class imbalance mitigation (via SMOTE), and supervised classification. Finally, we apply a heuristic algorithm to refine which single UE-VBS is selected per cluster, subject to constraints like distance from the base station and minimum data rate. Throughout these phases, from data creation to model deployment, our objective is to identify eligible UEs in a dynamic network environment efficiently and reliably.

A. Dataset Generation

This subsection provides a thorough description of the dataset used for UE clustering, classification, and imbalance handling. It details how UE positions are generated, their mobility is modeled, and various parameters (e.g., memory, battery discharge rate, SINR) are computed. Additionally, the section addresses class imbalance using oversampling techniques like SMOTE to ensure robust and equitable model performance.

[1]The number of ineligible UEs is more than the number of eligible UEs.

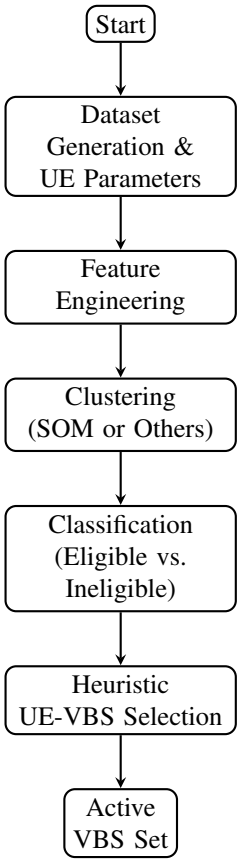


Fig. 1: Flowchart of the proposed approach, illustrating the stages from dataset generation and feature engineering to clustering, classification, and final UE-VBS selection.

1) Clustering: The UE positions are initialized as Cartesian pairs, i.e., (x, y) coordinates in space. A random initialization of angle, speed, and time intervals is used to facilitate a dynamic selection environment, allowing UEs to move. Because UEs are mobile nodes, their positions and movements change at various intervals. Specifically, the speed is varied randomly between 0 m/s and 10 m/s, while the angle ranges from 10° to 350°. These parameters change every 20 s (e.g., 20 s, 40 s, 60 s, etc.). The simulation models a single BS located at the center of a $1000\,\text{m} \times 1000\,\text{m}$ area, representing the coverage zone. UEs are distributed non-uniformly around the BS to mimic realistic user distributions commonly found in urban and suburban environments. This distribution follows a radial gradient, with higher UE densities near the BS and gradually decreasing densities outward. The UE distribution is modeled using a Poisson Point Process (PPP) with a density function

$$\lambda(r) = \lambda_0 e^{-\alpha r},$$

where λ_0 is the UE density at the BS location, and α governs the rate at which density decreases with distance from the

BS. The simulation is run for multiple population densities, as shown in Fig. 2, and the required parameters are recorded. Then, at a random time, a snapshot of the coordinates is captured and sent to the clustering algorithm to perform UE clustering at that specific moment.

2) Classification for Eligible UE: To build a robust dataset for clustering, we generate multiple snapshots of UE distributions with populations ranging from 100 to 1000 in increments of 100. Each snapshot represents a distinct network scenario, thereby enabling a comprehensive analysis across various network loads and densities. This approach ensures the classification model is trained and tested under diverse conditions, enhancing its generalizability. UEs are characterized by both **randomly assigned** and **computed** parameters, as follows:

- **Randomly Assigned Parameters**:
  - `memory`: Represents the device's RAM, randomly assigned between 2 GB and 16 GB to reflect diverse device capabilities.
  - `instantaneous_battery_percentage`: Indicates the current battery level, randomly assigned between 10% and 100%, simulating real-time battery states.
  - `cpu_power`: Reflects the device's processing capability, randomly assigned between 1 GHz and 3 GHz to simulate different performance levels.
- **Computed Parameters**:
  - `battery_discharge_rate`: Calculated using $\texttt{battery_discharge_rate} = \beta_1 \times \frac{\texttt{cpu_power}}{\texttt{memory}} + \beta_2 \times \frac{1}{\texttt{instantaneous_battery_percentage}}$, where β_1 and β_2 are empirically determined weighting factors. In a more detailed practical context, battery discharge can also be expressed as
  $$\text{Battery Discharge Rate} = \frac{\text{Capacity (mAh)}}{\text{Discharge Time (h)}}, \tag{7}$$
  where the capacity varies (e.g., 4000–7000 mAh) and discharge time ranges from 1 to 15 hours.
  - `distance_from_base_station`: Derived via the Euclidean distance formula:
  $$\sqrt{(x - x_{\text{BS}})^2 + (y - y_{\text{BS}})^2},$$
  where $(x_{\text{BS}}, y_{\text{BS}})$ are the coordinates of the BS (typically $(500\,\text{m}, 500\,\text{m})$).
  - `received_power`: Modeled by the Friis transmission equation:
  $$\texttt{received_power} = \frac{P_t\, G_t\, G_r\, \lambda^2}{(4\pi d)^2\, L},$$
  where P_t is the BS transmit power, G_t and G_r are the antenna gains, λ is the carrier wavelength, d is the UE–BS distance, and L represents system losses. In logarithmic form, path loss can also be written as
  $$[P_L(d)]_{\text{dB}} = [P_L(d_0)]_{\text{dB}} + 10\alpha \log_{10}\Big(\frac{d}{d_0}\Big) + \chi,$$
  where α is the path loss exponent, χ is a Gaussian random variable, and d_0 is a reference distance.
  - `SINR`: Calculated as
  $$\texttt{SINR} = \frac{\texttt{received_power}}{\texttt{interference_power} + N_0},$$
  where `interference_power` is the aggregate interference from other transmitting UEs, and N_0 is the thermal noise power given by $k_B T B$.

After obtaining the primary parameters, we also compute the following performance indicators:

1) **Data Rate**: Estimated using the Shannon capacity formula
$$R = B \log_2(1 + \text{SINR}), \tag{8}$$
where B is the channel bandwidth. The sum rate for a cluster of size c is:
$$TP = \sum_{i=1}^{c} B \log_2(1 + \text{SINR}_i). \tag{9}$$
2) **QoS**: Quantified using Jain's fairness index (JFI),
$$J(x_1, x_2, \ldots, x_{\bar{N}}) = \frac{\Big(\sum_{i=0}^{m} x_i\Big)^2}{\bar{N}\Big(\sum_{i=0}^{m} x_i^2\Big)},$$
where $\bar{N}$ is the number of users at a given time, and x_i represents the throughput for the ith connection. The spectral efficiency between a UE and its UE-VBS can be measured by
$$UE_i VBS_k = \sum_{\substack{n=1 \\ n \neq k}}^{N} \log_2\Big(1 + \frac{p_i^k\, h_i^k}{p_{VBS,j}^n\, h_{VBS,j}^n}\Big).$$
3) **QoE**: Evaluates perceived user quality. It is assessed via a QoE fairness index:
$$F = 1 - \frac{2\sigma}{H - L},$$
where L and H are the bounds of the rating scale, and σ is the standard deviation of measurements. The fairness score ranges from $\frac{1}{N}$ (worst) to 1 (best).

With all parameters computed (distance from the BS, battery discharge rate, received power, SINR, memory, CPU power, data rate, QoS, and QoE), the dataset is assembled for training, testing, and evaluating ML algorithms. A comprehensive brute-force approach, implemented in Python, generates and stores these parameters for diverse network conditions.

After clustering, each UE is inspected to determine its eligibility to act as a UE-VBS. In particular, we evaluate:

1) Battery discharge rate
2) Received power
3) SINR
4) UE memory
5) CPU power

Once these primary metrics are verified, additional indicators (data rate, QoS, and QoE) are also considered. Finally, data

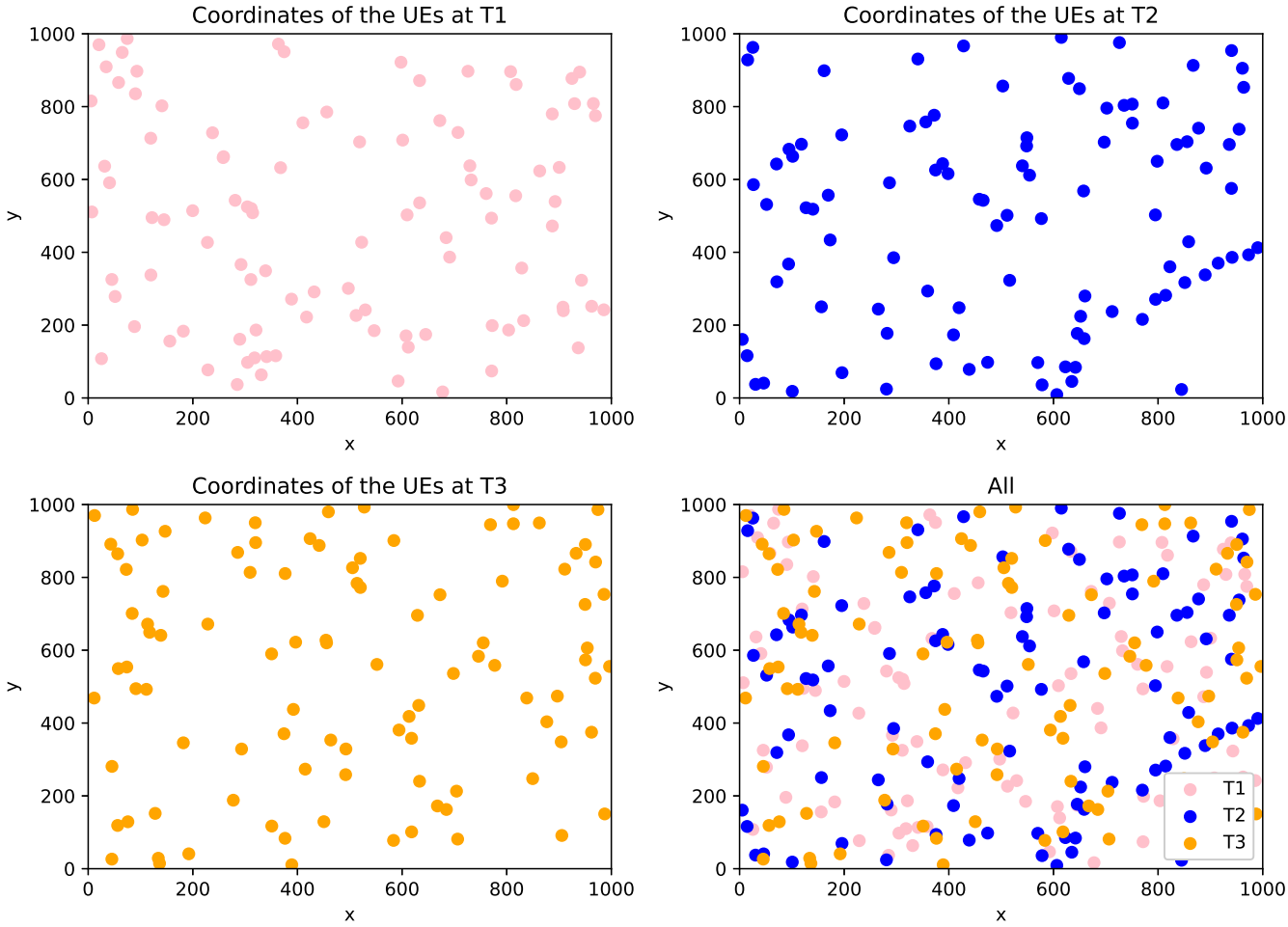


Fig. 2: Simulation of UE movement in the spatial and temporal domain.

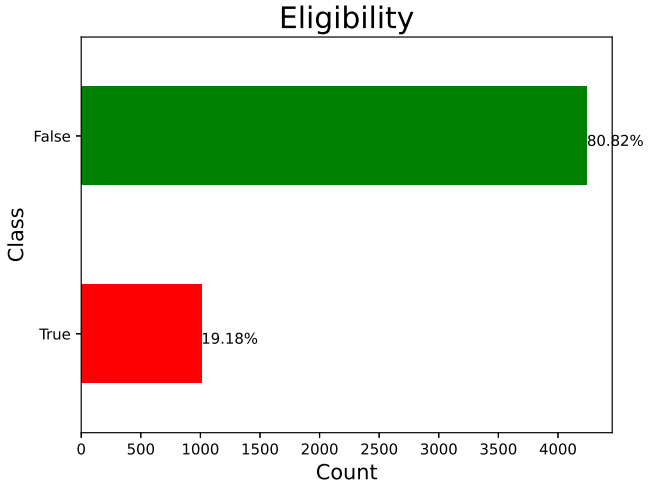
cleaning[2] is performed to ensure the integrity of the training data. This comprehensive process—spanning dataset generation, feature computation, and parameter evaluation—forms the basis for accurately classifying UEs as either eligible or ineligible for UE-VBS functionality.

3) Class Imbalance & Overcoming Imbalance: This subsection examines the strategy employed to address the class imbalance problem. Out of 5250 data points, only 1007 represent devices eligible to become UE-VBS, as illustrated in Fig. 3. This bias can lead to incorrect learning, known as the accuracy paradox, where accuracy is not a reliable metric for imbalanced datasets because it simply reflects the proportion of correctly classified instances. In classification problems, errors occur, and the target class is typically the focus of interest. Therefore, other reliable measures, such as precision and recall scores, are used to evaluate model performance. It is observed that models achieve higher recall and F1 scores for the negative class (class 0) compared to the positive class (class 1). The classifiers' poor performance on class 1 relative to class 0 is attributed to the inherent skew toward the class of devices that are 'ineligible' to become UE-VBS. To overcome this bias, we perform sampling to ensure equal representation of the two classes. There are two sampling methods to balance the dataset:

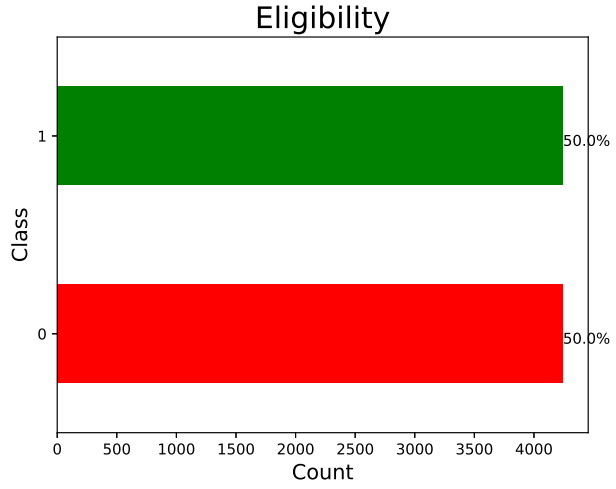
- Undersampling: Deleting samples from the majority class until the desired class distribution is achieved.
- Oversampling: Duplicating samples from the minority class until the desired class distribution is achieved.

The Synthetic Minority Over-sampling Technique (SMOTE) [9] is an oversampling method that operates by selecting examples that are near each other in the feature space, generating a line between these examples, and creating a new sample at a point along this line. SMOTE balances the training dataset

[2]Data cleaning removes or fixes incorrect, corrupted, or incomplete data within a dataset.



(a) Class distribution before SMOTE



(b) Class distribution after SMOTE

Fig. 3: Overcoming Class Imbalance using SMOTE

by synthesizing data points with smooth variation and high correlation with the existing data. As illustrated in Fig. 3b, the class distribution after applying SMOTE significantly reduces the class imbalance problem.

a) Handling Imbalance More Accurately: The issue of dataset imbalance is a significant challenge in many machine learning applications, particularly in fields such as network management and failure detection. Two approaches address this issue differently:

1) In [55], the authors address dataset imbalance using advanced machine learning techniques, particularly transfer learning and reinforcement learning. Transfer learning utilizes knowledge from related tasks to improve learning in imbalanced scenarios, leveraging pre-trained models on similar problems to address challenges posed by small or imbalanced datasets without compromising model performance. Reinforcement learning dynamically adjusts resource allocation to ensure that minority classes receive appropriate attention, promoting a more balanced distribution of network resources. This adaptive approach ensures that the model remains robust and performs well even when the data is imbalanced.
2) In [52], the authors tackle dataset imbalance through a hybrid feature selection method within a failure detection framework for IP core networks. This method combines multiple feature selection techniques to ensure that the most relevant and informative features are selected for model training. By focusing on the most significant features, the framework can mitigate the impact of imbalance in the dataset. This approach helps improve the accuracy and reliability of the failure detection system, even in the presence of imbalanced data. The hybrid feature selection method ensures that the model remains sensitive to minority class instances, enhancing the overall performance and detection capabilities in complex network environments.

In our approach, we follow the suggestions from the two referenced papers and combine their techniques to handle dataset imbalance effectively. To implement these approaches, we followed these steps:

1) **Data Standardization**: We standardized the features to ensure that each feature contributes equally to the model. This is crucial for handling imbalanced datasets, as it ensures that no single feature disproportionately influences the model's performance.
2) **Advanced Resampling Techniques**: We resampled the minority class to match the majority class size using advanced resampling techniques. This simulates the effects of transfer learning and reinforcement learning by ensuring that the model has enough data from the minority class to learn effectively.
3) **Hybrid Feature Selection Method**: Although we assumed all features are significant in our implementation, the hybrid feature selection method typically involves selecting the most relevant and informative features to improve model accuracy and reliability.
4) **Model Training and Evaluation**: We trained a RandomForestClassifier on the resampled, standardized data and evaluated its performance using a classification report. This report provides insights into the model's precision, recall, and F1-score, helping us assess its effectiveness in handling imbalanced data.

Combining these techniques, we aimed to create a robust model that performs well even with imbalanced datasets. The results demonstrate that our approach effectively addresses dataset imbalance, ensuring that the model remains accurate and reliable.

B. First Stage for Separation of UEs into Clusters

In this subsection, we outline the methodology used for clustering the devices under a BS. We have conducted a

comprehensive analysis and comparison of various clustering approaches to evaluate their effectiveness in organizing UEs based on proximity to the base station. The clustering algorithms are categorized into two groups: those that require prior knowledge of the number of clusters and those that do not. Specifically, we evaluated K-Means, Spectral Clustering, Agglomerative Clustering, and BIRCH, which require the number of clusters to be specified beforehand, as well as Mean-Shift, Affinity Propagation, OPTICS, and SOM, which do not need the number of clusters to be known a priori. This analysis identified the best-performing clustering techniques, paving the way for their integration into the proposed machine-learning pipeline.

Additionally, we provided a detailed description of the dataset generation process, feature selection, and experimental execution of these clustering algorithms. This framework/strategy offers a structured approach to assessing and comparing clustering methods, ensuring a thorough evaluation of their performance based on metrics like the silhouette score (as indicated in Section V-C). Through this work, we aim to identify the optimal clustering techniques for further stages of development in the pipeline.

Clustering involves organizing a set of data objects into groups or clusters such that objects within the same cluster are more similar to each other than to those in other clusters. In this first stage of the proposed machine learning pipeline, we focus on clustering UEs based on their distance from the base station. Clustering algorithms can be categorized based on whether they require prior knowledge of the number of clusters. In our work, we tested several clustering algorithms to determine the most effective method for clustering UEs. Specifically, the clustering algorithms considered are i) Mean-Shift, Affinity Propagation, Ordering Points To Identify the Clustering Structure (OPTICS) and SOM, which do not require the number of clusters to be known beforehand (as shown in Table IX at Section V-C); and ii) K-Means, Spectral Clustering, Agglomerative Clustering, Balanced Iterative Reducing and Clustering Hierarchies (BIRCH) which require the number of clusters to be specified beforehand (compared in Table VIII at Section V-C).

1) The First Stage Clustering Algorithm: To systematically evaluate the clustering algorithms, the following steps are undertaken in the experimental setup: The steps outlined in Algorithm 1 describe the procedure for conducting the clustering experiment. Initially, 1000 UEs are randomly generated within a predefined spatial area. The Euclidean distance matrix, which captures the pairwise distances between all UEs, is then computed.

Moreover, we tested algorithms that do not require the number of clusters to be known beforehand (e.g., Mean-Shift, Affinity Propagation, OPTICS, and SOM) by applying the clustering directly and computing the silhouette coefficient to measure cluster quality. For those methods that do require specifying the number of clusters, we employed Silhouette Analysis or the Elbow Method to determine an appropriate k.

Algorithm 1 Clustering Experiment Execution

Input: Number of UEs $N = 1000$, Clustering Algorithms $\mathcal{A} = \{K$-Means, Spectral Clustering, Agglomerative Clustering, BIRCH, Mean-Shift, Affinity Propagation, OPTICS, SOM$\}$.
Output: Best clustering algorithm based on silhouette score.
Generate N random UE positions (x, y) using a uniform random distribution.
Compute the Euclidean distance matrix D between all UE pairs.
for each algorithm $\alpha \in \mathcal{A}$ **do**
    if α requires the number of clusters k as input **then**
        Estimate the optimal number of clusters k^* using the Elbow Method or Silhouette Analysis.
        Apply α with k^* to the distance matrix D to form clusters.
    else
        Apply α directly to the distance matrix D to form clusters.
    end if
    Compute the silhouette score S_α for the resulting clusters.
    Store S_α for comparison.
end for
Determine the algorithm α^* with the highest silhouette score S_{α^*}.
Return α^* as the best clustering algorithm for the dataset.

For clustering methods that require the number of clusters to be predetermined (such as K-Means, Spectral Clustering, Agglomerative Clustering, and BIRCH), it is necessary to estimate the optimal cluster count. This estimation is commonly performed using the Elbow Method—which examines the within-cluster sum of squares (WCSS)—or via Silhouette Analysis, which identifies the number of clusters that maximizes the silhouette score. In our approach, we employ Silhouette Analysis [50] (illustrated in Fig. 4) to determine the ideal value for the number of clusters, $\bar{N}$. The silhouette score measures how similar each data point is to its cluster compared to other clusters. Specifically, for a given sample, let i be its mean intra-cluster distance (the average distance to other points in the same cluster) and n be its mean nearest-cluster distance (the average distance to points in the closest neighboring cluster). The Silhouette coefficient is then defined as

$$\frac{n - i}{\max(i, n)}, \tag{10}$$

which ranges from -1 (poor assignment) to $+1$ (ideal separation). In typical silhouette plots, cluster labels appear on the Y-axis and the silhouette score on the X-axis; the thickness of each silhouette reflects the number of samples in that cluster. A high average silhouette coefficient (close to $+1$)

indicates that clusters are well separated, whereas a score near 0 suggests that some points lie on the decision boundary, and negative values imply possible misclassification. As shown in the example Fig. 4, the average silhouette score peaks when $\bar{N} = 4$ (compared to 2, 3, or 5), with the global score being slightly over 0.4, thus indicating four as the optimal number of clusters.

Finally, the quality of the resulting clusters from each algorithm is assessed using the silhouette score. The algorithm with the highest silhouette score is selected as the most suitable for this particular dataset and use case.

2) Complexity Analysis of the Clustering Approaches: The results from Table III compare clustering algorithms that estimate the number of clusters, including Mean-Shift, Affinity Propagation, OPTICS, SOM, and ClusterGAN. Mean-Shift and Affinity Propagation use Euclidean distance as their metric, while OPTICS uses Minkowski distance for increased flexibility in distance calculations. SOM also uses Euclidean distance but is optimized for preserving the topology of data on a lower-dimensional grid. ClusterGAN leverages a generative adversarial network (GAN) framework to learn latent representations and perform clustering in an unsupervised manner. In terms of time complexity, Mean-Shift, and Affinity Propagation have a complexity of $O(n^2t)$ (where t represents the number of iterations) [12], whereas OPTICS and SOM are slightly more efficient at $O(n^2)$ [36]. ClusterGAN, as a deep learning–based method, also has a time complexity of $O(n^2t)$ [44]. SOM's complexity of $O(n^2)$ makes it scalable for medium-sized datasets. Moreover, Mean-Shift, SOM, and OPTICS terminate deterministically, while Affinity Propagation and ClusterGAN use iterative termination methods, rendering their convergence behavior less predictable. In contrast, Table IV presents a complexity comparison of popular clustering algorithms—such as k-Means, Spectral Clustering, Agglomerative Clustering, and BIRCH—that require the number of clusters (k) to be specified. These methods differ in their time complexities and scalability depending on factors like distance metrics and linkage criteria.

C. Second Stage of Eligible UEs Selection

This subsection outlines the methodology used in the second stage of the machine learning process, where various classification techniques are analyzed and compared to assess their effectiveness in identifying eligible UEs to function as UE-VBS. The process begins with the generation and preparation of the dataset, including selecting relevant features that are critical for accurate classification. Next, it describes the training and evaluation of different classification models, addressing key challenges such as class imbalance and the importance of hyperparameter tuning to enhance model performance. This framework/strategy offers a structured approach for systematically comparing the models using performance metrics, ensuring that the most effective classification approach is selected for the task of UE-VBS eligibility determination (as indicated in Section V-D).

Classification is a supervised process in which a classifier is trained on pairs of input data and the corresponding desired output. The classifier aims to predict the correct class or value based on valid input data. In our work, classification constitutes the second stage of the machine learning engine, aiming to categorize the UEs within the clusters as either eligible or ineligible to become UE-VBS. The UEs are assessed based on the metrics discussed in Section IV-C0a. For this analysis, the target variable 'eligibility' is set to 1 for eligible UEs and 0 for ineligible UEs. The dataset is split into 70% for training and 30% for testing, and the performance of different models is studied and compared using the performance metrics detailed in Subsection V-D1.

a) Feature Selection: We employ several feature selection techniques to identify the most influential variables for the classification stage of selecting eligible UEs to act as UE-VBSs. Initially, features such as battery discharge rate, received power, SINR, memory, and CPU power are considered. To further enhance model performance, additional features like data rate, QoS, and QoE are also evaluated. The following approaches are utilized:

- **Fisher Scores [28]:**
  This method evaluates the discriminative power of each feature by computing the ratio of the variance between classes to the variance within classes. The Fisher Score is given by:
  $$F_j = \frac{\sum_{i=1}^{c} n_i(\mu_{ij} - \mu_j)^2}{\sum_{i=1}^{c} n_i \sigma_{ij}^2} \tag{11}$$
  where F_j is the Fisher Score for the j-th feature, c is the number of classes, n_i is the number of samples in the i-th class, μ_{ij} is the mean of the j-th feature in the i-th class, μ_j is the overall mean of the j-th feature, and σ_{ij}^2 is the variance of the j-th feature in the i-th class.
- **Mutual Information Scores [17]:**
  This approach quantifies the amount of shared information between a feature and the target variable, calculated as:
  $$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log \left(\frac{p(x,y)}{p(x)p(y)} \right) \tag{12}$$
  where $I(X;Y)$ is the mutual information between feature X and target Y.
- **Chi-Square Scores [33]:**
  The Chi-Square test measures the dependence between a feature and the target variable by comparing observed and expected frequencies:
  $$\chi^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i} \tag{13}$$
  where O_i is the observed frequency, E_i is the expected frequency, and n is the number of categories or classes.

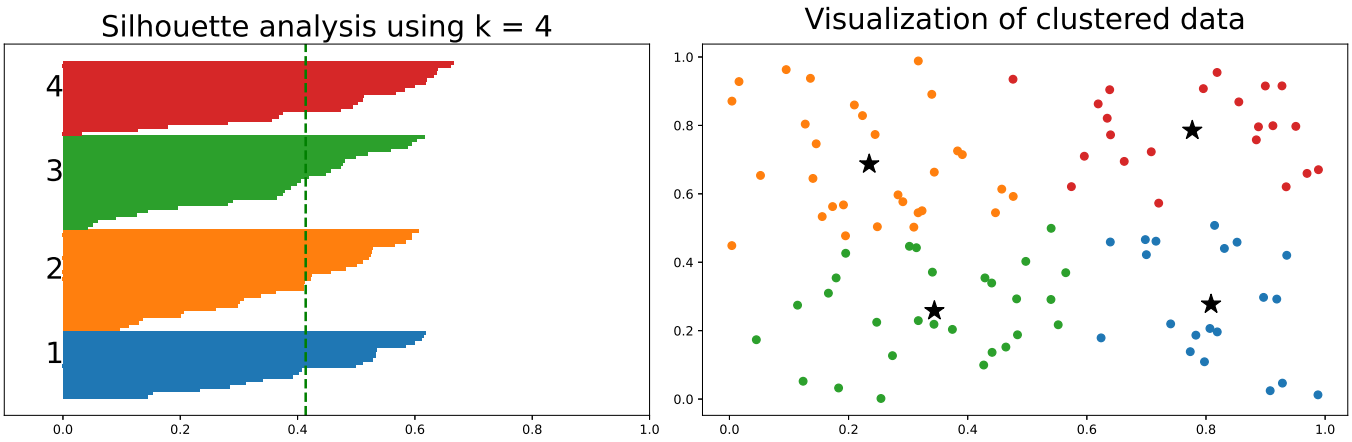


Fig. 4: Silhouette plot of $k = 4$ (best fit) for K-Means Clustering.

TABLE III: Complexity Comparison between different clustering algorithms that estimate the number of clusters

Clustering Algorithm	Parameters	Metric	Time Complexity	Scalability	Termination Condition
Mean-Shift	Kernel Bandwidth	Euclidean Distance	$O(n^2t)$	Scalable	Deterministic
Affinity Propagation	Damping Factor, Sample Preference	Euclidean Distance	$O(n^2t)$	Not Scalable	Not Deterministic
OPTICS	Minimum Cluster Membership	Minkowski Distance	$O(n^2)$	Scalable	Deterministic
SOM	Learning Rate, Grid Size	Euclidean Distance	$O(n^2)$	Scalable	Deterministic
ClusterGAN	Latent Code, Generator, Discriminator, Clustering Regularization	Learned Latent Space	$O(n^2t)$	Moderate	Iterative (via adversarial training)

TABLE IV: Complexity Comparison between different clustering algorithms that require the number of clusters as a parameter

Clustering Algorithm	Parameters	Metric	Time Complexity	Scalability	Termination Condition
K-Means	Number of clusters k, initialization	Euclidean Distance	$O(n^2)$	Moderate	Deterministic
Spectral Clustering	Number of clusters k, affinity matrix	Euclidean Distance	$O(n^3)$	Low	Deterministic
Agglomerative Clustering	Number of clusters k, linkage criterion	Euclidean Distance	$O(n^2)$	Moderate	Deterministic
BIRCH	Number of clusters k, branching factor	Euclidean Distance	$O(n)$	High	Deterministic

- **Random Forest Feature Importances [18]:**
  In a Random Forest model, the importance of a feature is determined by the reduction in impurity (e.g., Gini impurity) it provides:

$$FI = \sum_{t \in T} p(t) \cdot \Delta i(s,t) \tag{14}$$

  where FI is the feature importance, t represents each node in the set of trees T, $p(t)$ is the proportion of samples reaching node t, and $\Delta i(s,t)$ is the decrease in impurity at node t attributed to feature s.

The feature selection process for classifying eligible UEs directly addresses the challenges discussed in Section III-A. In the context of spectrum availability, the increased number of small cells, the complexity of 5G networks, and energy consumption concerns, features such as battery discharge rate, received power, SINR, memory, CPU power, QoS, QoE, and data rate are selected due to their high relevance in optimizing network performance. By including all these features in the training (and testing) process (commonly using a 70/30 split), our model is well-equipped to adapt to the complexities and demands of 5G networks while maintaining high service quality and operational efficiency. Table V presents the selected features for machine learning training, with their respective scores across Fisher Score, Mutual Information, Chi-Square, and Random Forest Importance metrics. Figure 5 illustrates the feature evaluation scores across the different metrics.

In summary, this unified approach to dataset generation and systematic feature selection—employing multiple evaluation techniques—ensures that the most relevant variables are retained for the classification of eligible UEs as UE-VBSs. The inclusion of traditional parameters alongside enhanced features such as data rate, QoS, and QoE significantly enhances the model's predictive performance, leading to improved accuracy and robustness in a dynamic 5G environment.

1) Common Characteristics of the Investigated Approaches: The classifiers considered in our work (i.e., Random Forest, Adaptive Boosting, Gradient Boosting) use decision tree models. Decision trees consist of decision nodes that split the data. Each node represents a question that helps arrive at a final decision, denoted by the leaf node. Decision trees seek to find the best split to divide the data, and they are typically trained through the Classification and Regression Tree (CART) algorithm. Moreover, the approaches investigated use

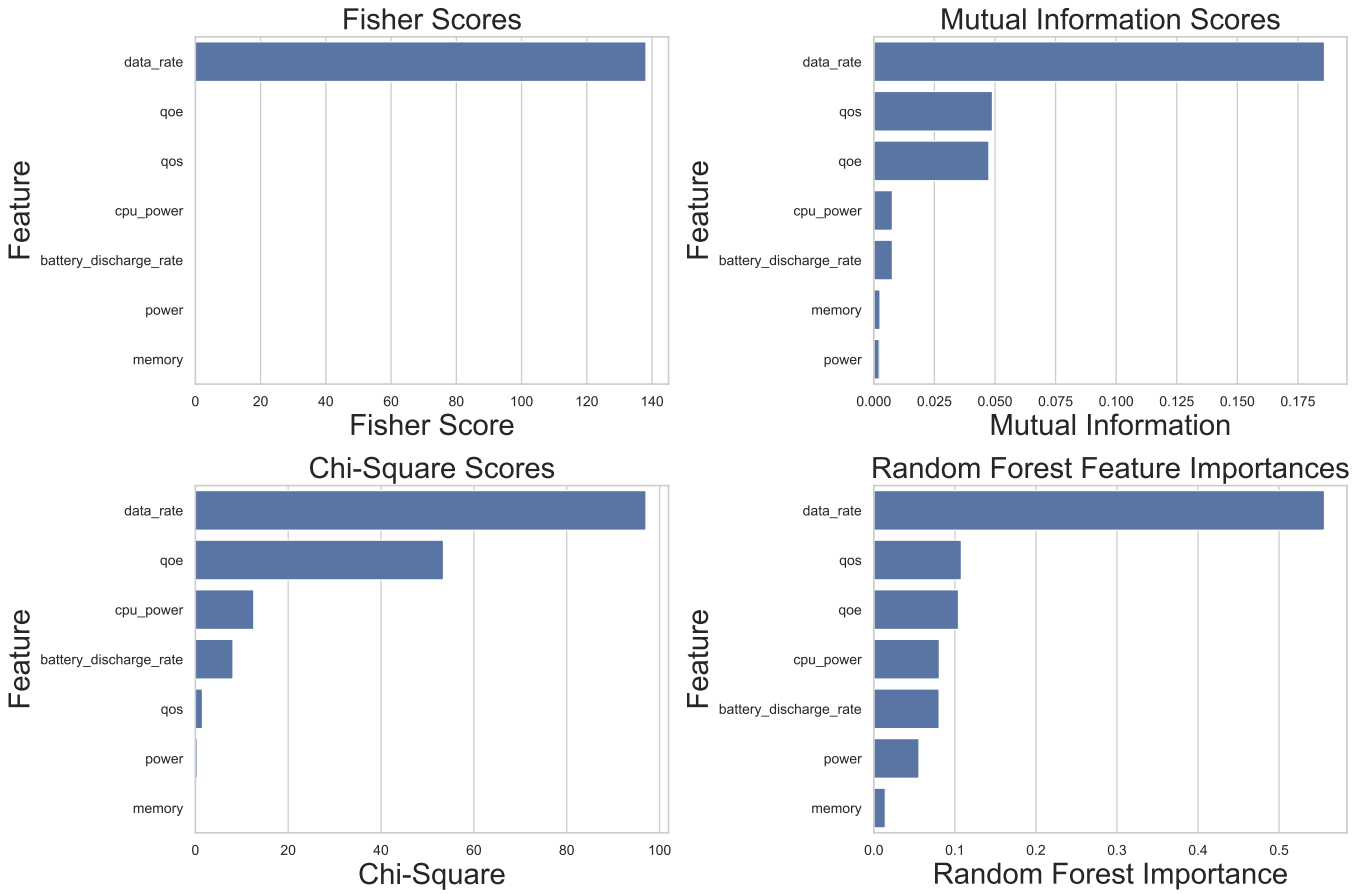


Fig. 5: Feature evaluation scores across different metrics: Fisher Scores, Mutual Information Scores, Chi-Square Scores, and Random Forest Feature Importances.

TABLE V: Selected Features for ML Training

Feature	Fisher Score	Mutual Information	Chi-Square	Random Forest Importance
battery_discharge_rate	0.038989	0.007662	8.121053	0.083747
power	0.004150	0.002228	0.399056	0.054644
memory	0.000004	0.000562	0.001386	0.014847
cpu_power	0.063115	0.007674	12.593612	0.083912
QoS	0.127560	0.048961	1.535783	0.098501
QoE	0.169708	0.047493	53.459816	0.112582
data_rate	138.146665	0.186285	97.079046	0.551767

ensemble learning methods, which are composed of a set of decision tree classifiers[3]. Their predictions are aggregated to determine the most popular result. By combining several individual classification models, ensemble models [16] tend to be more flexible (less biased) and less data-sensitive (less variance). The two popular ensemble methods are bagging and boosting, briefly explained below:

- **Bagging**, also known as bootstrap aggregation, involves training multiple individual models in parallel, with each model trained on a random subset of the data selected from the training dataset with replacement[4].
- **Boosting**: involves training individual models sequentially, with each model learning from the mistakes made by the previous model.

2) The Second Stage of the Classification Algorithm: To systematically evaluate the classification algorithms, the

[3]Although decision trees are commonly used, other models like Support Vector Machines (SVM) can be employed.

[4]This means that individual data points can be selected more than once.

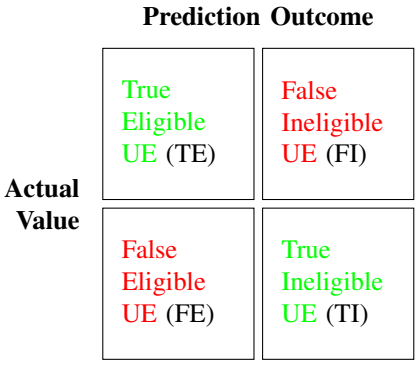


Fig. 6: Confusion Matrix.

following steps are undertaken in the experimental setup as shown in the Algorithm 2:

Algorithm 2 Classification Experiment Execution

Input: Dataset $\mathcal{D}$ with features $\mathbf{X}$ and labels $\mathbf{y}$, Classification Algorithms $\mathcal{C} = \{$Random Forest, AdaBoost, Gradient Boosting$\}$.
Output: Best classification algorithm based on F1-score.
Split $\mathcal{D}$ into training set $\mathcal{D}_{train}$ (70%) and test set $\mathcal{D}_{test}$ (30%).
Standardize the features in $\mathbf{X}$ to ensure each feature has zero mean and unit variance.
Address class imbalance in $\mathcal{D}_{train}$ using SMOTE.
for each classifier $\gamma \in \mathcal{C}$ **do**
    if γ requires hyperparameter tuning **then**
        Perform cross-validated grid search to identify optimal hyperparameters for γ.
    end if
    Train γ on $\mathcal{D}_{train}$ using the identified or default hyperparameters.
    Evaluate γ on $\mathcal{D}_{test}$ using metrics such as accuracy, precision, recall, and F1-score.
    Store F1-score F_γ for comparison.
end for
Determine the classifier γ^* with the highest F1-score F_{γ^*}.
Return γ^* as the best classification algorithm for the dataset.

3) Complexity Analysis of Classification Approaches: The proposed solution employs several supervised classification algorithms to determine the eligibility of UEs as UE-VBSs. Table VI summarizes the key parameters, training time complexity, scalability, and termination conditions for the classifiers used in our study.

Here, n represents the number of training instances, m the number of features, and T the number of trees or boosting rounds. The Random Forest algorithm typically has a training complexity of $O(T \cdot n \log n \cdot m)$ due to the cost of building each tree on a bootstrap sample [8]. Boosting methods such as AdaBoost and Gradient Boosting usually have a complexity of $O(T \cdot n \cdot m)$ per boosting round [20], [22]. During prediction, each model requires $O(T \cdot d)$ time per instance, where d is the average depth of the decision trees.

D. Heuristic Algorithm for Selecting UE-VBSs from the Eligible UEs

This section provides the heuristic algorithm, which we implemented and used to select the UE-VBS from the eligible UEs, one per cluster after clustering is carried out at the first stage and classification in the second stage. The heuristic algorithm is provided in Algorithm 3. The heuristic algorithm selects the UE-VBS from the eligible UEs in a cluster (for generality, we named the selected cluster *C*). More specifically, the selection process uses two thresholds: i) the maximum distance allowed between the eligible UEs and the BS, and ii) the minimum achievable data rate between the eligible UE and the BS. The need for heuristic methods arises due to the challenges in selecting the optimal UE-VBS. The two-stage machine learning approach, consisting of clustering and classification, is used to identify eligible UEs based on factors like proximity to the base station, QoS, data rates, and battery life. However, the final selection of the UE-VBS requires additional refinement to address issues such as maximizing data rates and maintaining energy efficiency. The heuristic algorithm addresses these challenges by using thresholds for maximum distance from the base station and minimum data rate, ensuring the most suitable UE-VBS is selected, even if it is not the closest device. This approach is critical for refining the selection and dynamically adapting to network conditions while optimizing energy use and ensuring high throughput.

Moreover, it briefly explains the algorithm shown in Algorithm 3. Initially, the algorithm receives as parameters the cluster that will find the UE-VBS, the eligible UEs set related to the cluster with some additional information about each UE such as the data rate, distance to the BS, battery discharge rate, and the maximum distance allowed between the UEs and the BS. As the first step, the algorithm filters the eligible UEs and selects those closer to the BS than the threshold. After that, the UE from the eligible ones with the lowest battery discharge rate and that achieves the minimum data rate required is selected. Finally, if there are UEs within the cluster, and if no UE satisfies the above conditions, then the heuristic algorithm selects the one with the maximum distance allowed, a minimum battery discharge rate, and the highest data rate achieved.

1) Complexity Analysis of Dual-Stage ML Heuristic Framework: The proposed solution integrates three main components: (i) a clustering stage, (ii) a classification stage, and (iii) a heuristic algorithm for the final UE-VBS selection. Let n denote the total number of UEs and m the number of features. Depending on the chosen algorithm, the clustering complexity can vary. For example, SOM exhibit a worst-case complexity of $O(n^2)$ [36], while K-Means (with parameter tuning) also has a worst-case complexity of $O(n^2)$; alternatively, methods such as BIRCH can achieve $O(n)$ complexity, offering higher scalability for large n. After clustering, each cluster's UE data

TABLE VI: Comparison between different classification algorithms used for UE-VBS eligibility determination

Classification Algorithm	Random Forest	AdaBoost	Gradient Boosting
Parameters	Number of trees T, maximum depth, minimum samples per split	Number of estimators T, learning rate	Number of estimators T, learning rate, maximum depth
Metric	Gini impurity, information gain	Exponential loss	Loss function minimization (e.g., squared error, logistic loss)
Training Time Complexity	$O(T \cdot n \log n \cdot m)$ [8]	$O(T \cdot n \cdot m)$ [20]	$O(T \cdot n \cdot m)$ [22]
Scalability	Moderate-to-High (parallelizable)	Moderate	Moderate
Termination Condition	Deterministic (fixed number of trees)	Iterative (stops when error converges or after T iterations)	Iterative (stops upon convergence of the loss function)

Algorithm 3 Heuristic Algorithm to Select the UE-VBS in Cluster C

Input: Cluster C, Eligible UE set UE_{set}, max distance to BS d_{max}, minimum data rate R_{min}
Output: Selected UE-VBS UE_{VBS}

$candidates \leftarrow \{ue \in UE_{set} \mid dist(ue, BS) \leq d_{max}\}$
if $candidates \neq \emptyset$ **then**
    $filtered_candidates \leftarrow \{ue \in candidates \mid R(ue, BS) \geq R_{min}\}$
    $UE_{VBS} \leftarrow \arg\min_{ue \in filtered_candidates} battery_discharge_rate(ue)$
else
    $UE_{VBS} \leftarrow \arg\min_{ue \in UE_{set}} \{dist(ue, BS), battery_discharge_rate(ue)\}$
end if
return UE_{VBS}

is classified using methods such as Random Forest, whose complexity is approximately $O(T \cdot n \log n \cdot m)$ [8]. The final selection of a UE-VBS within each cluster is performed by a heuristic algorithm that iterates over candidate UEs; assuming there are k clusters and each cluster contains roughly n/k UEs, the selection in each cluster takes $O(n/k)$ time, leading to an overall complexity of $O(n)$ for all clusters. Summing the contributions of each stage, the total computational complexity of the proposed solution is approximately

$$O(n^2) + O(T \cdot n \log n \cdot m) + O(n).$$

In large-scale scenarios where n is very large (e.g., thousands or more UEs), the clustering stage (with quadratic complexity) may dominate the overall cost; however, by employing more scalable clustering methods (e.g., BIRCH with $O(n)$) or leveraging parallel processing, the overall computational load can be mitigated. In practice, even though the worst-case complexity appears high, practical optimizations enable the solution to scale effectively with increasing UE densities.

V. Evaluation

In this section, we have implemented the steps described in the methodology Section IV, spanning the entire data pipeline (from dataset generation and feature engineering to clustering, classification, and final UE-VBS selection), in our Matlab environment and structured our analysis around three sets of experiments. The first experiment focuses on the results of the clustering process conducted in the first stage of the methodology. The second experiment considers the classification process performed in the methodology's second stage. Finally, the third experiment analyzes the outcomes obtained from the heuristic algorithm introduced in Section IV-D, in conjunction with the Two-Stages ML approaches, applied after clustering and classification to select a single eligible UE per cluster as the UE-VBS. Moreover, this section presents, discusses, and scrutinizes the results produced by our methodology. We begin by examining the clustering phase, the initial step of the methodology. Next, we detail and interpret the results concerning the classification phase, which constitutes the methodology's second step. Lastly, we present and evaluate the findings related to the heuristic algorithm for selecting the UE-VBS from the eligible UEs within each cluster.

A. Simulation Setup

In this section, we describe the experimental setup involving the simulation of the UE-VBSs in a 5G network and the dataset generation for the classifier. Moreover, this section explains the various parameters and their corresponding analytical expressions used as features for classifying eligible and ineligible UEs. The simulations were performed on the machine described in Section V-A. The experimental setup is simulated using a Python environment and implemented to resemble a real-world scenario for extracting the required parameters for the classification task. The UEs are spread in a non-uniform distribution around the base station.

B. Machine Configuration

For all simulations in this work, we used a computing machine with the following specifications:

- **Processor:** Intel Core i5-8265U
- **Storage:** 256GB M.2 PCIe NVMe SSD
- **Memory:** 8GB DDR4 RAM (2400 MHz)
- **GPU:** NVIDIA GeForce MX150 (2GB GDDR5)

1) Simulation Parameters: To provide a clear overview of the key parameters used in our simulation, we present them in Table VII. This table includes both the hardware specifications of the computing machine and the critical simulation parameters that influence the behavior and performance of the UE-VBSs within the 5G network. In our experiment, the UEs are spread in a non-uniform distribution around the base station. Multiple snapshots of the UEs are taken with different population densities to provide a more comprehensive picture of real-world scenarios. The parameters of UEs such as `memory`, `instantaneous_battery_percentage`, and `cpu_power` were assigned randomly, while parameters like `battery_discharge_rate`, `distance_from_base_station`, `received_power`, and `signal interference noise ratio (SINR)`

are computed using the expressions presented in the subsequent section for all UE data points.

C. Comparative Evaluation of Clustering Results

This subsection provides a comparative evaluation of various clustering algorithms based on their complexity, scalability, and performance metrics (e.g., silhouette score). Two categories of clustering algorithms are discussed: those that estimate the number of clusters and those that require the number of clusters as a parameter. For algorithms estimating the number of clusters, Mean-Shift, Affinity Propagation, OPTICS, SOM, and ClusterGAN are analyzed. Notably, SOM offers the best silhouette score of 0.64, thanks to its ability to preserve the topological relationships within the data. Mean-Shift also provides scalability and deterministic termination, while Affinity Propagation is less predictable. ClusterGAN, a deep learning–based method, achieves a silhouette score of 0.45. For algorithms requiring the number of clusters, such as K-Means, Spectral Clustering, Agglomerative Clustering, and BIRCH, K-Means attains a silhouette score of 0.46 but necessitates additional computational steps to determine the optimal number of clusters. Based on these results, SOM emerges as the most effective clustering method for large and dynamic datasets—particularly in terms of accuracy and scalability.

Based on the silhouette scores[5] calculated for each approach (see Tables VIII and IX), SOM provides the best score among all the approaches considered, with a silhouette score of 0.64. Although Mean-Shift ranks lower with a score of 0.43, it has the advantage over K-Means, Spectral Clustering, Agglomerative Clustering, and BIRCH in that it does not require the number of clusters to be specified beforehand. SOM not only offers the highest silhouette score but also effectively preserves the topological relationships in the data, making it the most suitable approach for clustering complex, non-linear datasets (as shown in Fig. 7).

In addition to these traditional methods, we performed experiments using a deep learning–based clustering approach, ClusterGAN [43]. Our experiments indicate that although ClusterGAN—an unsupervised method that automatically estimates the number of clusters—obtains a moderate silhouette score of 0.45 and a cluster purity of 0.78, it falls short of SOM, which achieves a silhouette score of 0.64 and a purity of 0.82. Moreover, the average computational time using SOM was 15.2 s compared to 42.7 s using ClusterGAN. These results suggest that for our problem—characterized by Euclidean spatial features and the need for real-time operation—SOM not only preserves the topological relationships in the data better but also demands substantially less computational overhead. Therefore, our decision to employ SOM is justified despite the existence of more recent deep learning–based methods.

[5]Silhouette scores are used both to identify the ideal number of clusters by performing a brute-force investigation using different numbers of clusters and to evaluate the clustering performance using the formula described in Eq. (10).

Furthermore, in addition to clustering quality metrics such as the silhouette score, it is important to assess the computational efficiency of each algorithm, particularly for large-scale network applications. For example, SOM required an average computational time of 15.2 seconds per run, whereas the deep learning–based ClusterGAN took approximately 42.7 seconds. Similar runtime experiments for the algorithms that require a predefined number of clusters showed that K-Means needed additional time (approximately 12.5 seconds on average, including the parameter tuning step), while BIRCH is the most efficient with a near-linear runtime in $O(n)$.

Table X summarizes both the average runtimes and approximate memory usage (scaled to simpler indicative values) at 1000 UEs for the key clustering methods. From these results, we observe that Mean-Shift and Affinity Propagation each consume about 42–43 MB of memory, reflecting their quadratic complexity for both time and space. BIRCH remains the most memory-efficient at only 20 MB, although it can sometimes yield lower-quality clusters for highly non-linear data. SOM uses roughly 30 MB, striking a practical balance between memory usage and performance. ClusterGAN exhibits the highest memory usage at about 50 MB, likely due to the overhead from training its generator and discriminator networks. Consequently, SOM not only provides superior clustering quality but also offers a feasible trade-off between runtime and memory needs for real-time network operations.

Overall, the computational efficiency analysis—encompassing both runtime and approximate memory consumption—complements the performance metrics, reinforcing the suitability of SOM for large-scale and real-time network clustering tasks.

As demonstrated above, SOM is the best approach for clustering our devices into clusters in the first stage, with a silhouette score of 0.6. Therefore, we select SOM as our first-stage machine-learning approach. Overall, from the results discussed, it is evident that SOM, followed by K-Means and Mean-Shift, yields the best results regarding the silhouette score. While K-Means offers good performance with a silhouette score of 0.46, it requires an additional step in the clustering process to determine the number of clusters beforehand, which adds computational overhead (in terms of CPU, memory, and time). In contrast, SOM not only avoids the need to predefine the number of clusters but also better preserves the topological structure of the data, making it a more suitable option for complex, dynamic datasets.

1) Clustering among SOM and K-Means: Figure 7 illustrates the distinct ways in which SOM and K-Means partition the same dataset. Lines 1 and 2 in the figure demarcate the boundaries between clusters identified by K-Means, which aims to minimize within-cluster variance. In contrast, SOM preserves the topological relationships among data points, resulting in smoother transitions across adjacent clusters, as can be seen by the gradual color changes in the map. This comparison highlights how different algorithmic strategies can lead to notably different clustering outcomes, even on the same dataset.

TABLE VII: Simulation Parameters

Parameter	Description	Value	Units
Processor	CPU model used for simulations	Intel Core i5-8265U	-
Storage	Type and capacity of SSD	256GB M.2 PCIe NVMe SSD	-
Memory	RAM specifications	8GB DDR4	2400 MHz
GPU	Graphics processing unit	NVIDIA GeForce MX150	2GB GDDR5
Carrier Frequency	Frequency band used	28	GHz
Transmit Power (P_t)	Base Station transmit power	20	dBm
Antenna Gain (G_t, G_r)	Transmitter and receiver antenna gains	15	dBi
System Losses (L)	Path loss and system-related losses	2	dB
Bandwidth (B)	Network bandwidth	100	MHz
Thermal Noise Power (N_0)	Calculated as $N_0 = k_B T B$	$1.38 \times 10^{-23} \times 290 \times 100 \times 10^6$	W
UE Density Function	Poisson Point Process density function	$\lambda(r) = \lambda_0 e^{-\alpha r}$	UEs/m^2
Density at BS (λ_0)	UE density at Base Station location	0.005	UEs/m^2
Decay Rate (α)	Rate of density decreases with distance	0.001	m^{-1}
UE Memory (`memory`)	Available RAM in UE	2–16	GB
Battery Percentage (`instantaneous_battery_percentage`)	Current battery level of UE	10–100	%
CPU Power (`cpu_power`)	Processing capability of UE	1–3	GHz

TABLE VIII: Comparison between different clustering algorithms that require the number of clusters as a parameter based on their silhouette score.

Clustering Algorithm	Silhouette Score
K-Means	0.46
Spectral Clustering	0.43
Agglomerative Clustering	0.44
Birch	0.44

TABLE IX: Comparison of various clustering algorithms that can estimate the number of clusters based on their silhouette score.

Clustering Algorithm	Silhouette Score
Mean-Shift	0.43
Affinity Propagation	0.43
OPTICS	0.24
SOM	0.64
ClusterGAN	0.45

D. Comparative Evaluation on Classification Results

This subsection provides a comparative evaluation of classification results, starting with the key performance metrics, followed by ROC curve evaluation, and concluding with classification results, model selection, validation, and explainability. We first introduce the primary metrics used to assess the models' effectiveness, including precision, recall, F1-score, and AUC-ROC. These metrics offer a detailed understanding of the model's predictive performance. Next, we conduct a thorough ROC curve evaluation to examine the trade-off between true positive and false positive rates. The AUC (Area Under the Curve) is used to measure the model's capability to distinguish between classes, with higher AUC values indicating better overall performance. After the ROC evaluation, we present the classification results for each tested model.

TABLE X: Average Runtime and Approx. Memory Usage of Clustering Algorithms (1000 UEs)

Clustering Algorithm	Average Runtime (s)	Approx. Memory (MB)
Mean-Shift	23.5	42
Affinity Propagation	28.1	43
OPTICS	19.0	28
SOM	15.2	30
ClusterGAN	42.7	50
K-Means (with tuning)	12.5	25
Spectral Clustering	35.0	40
Agglomerative Clustering	30.0	32
BIRCH	8.0	20

The results are compared across classifiers such as Random Forest, Gradient Boosting, and AdaBoost, highlighting their performance before and after addressing class imbalance using SMOTE. Random Forest consistently achieved superior results, with an accuracy of 0.97, and improved recall and F1 scores, particularly after SMOTE was applied. Finally, the examined classification algorithms were tested using K-Fold, Hold-out Testing, and Nested Cross-Validation to ensure a rigorous model selection process. Techniques like SHAP, LIME,

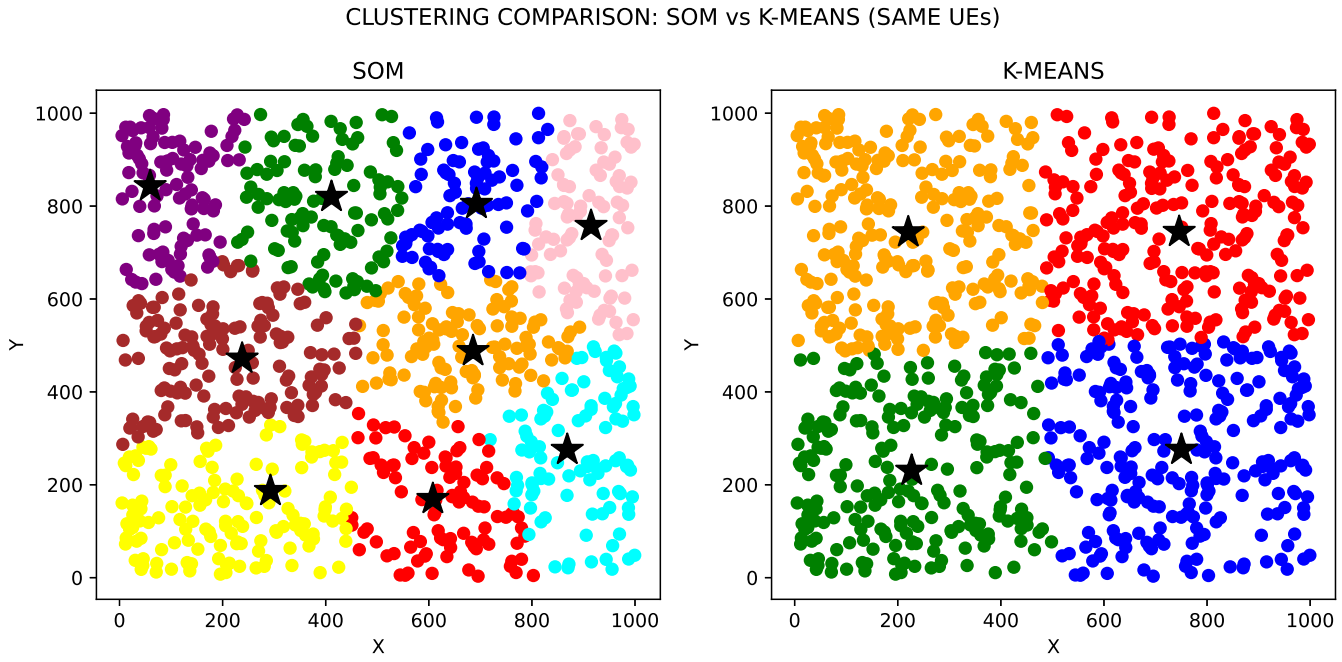


Fig. 7: Comparison of the clustering results of SOM and K-Means.

QLattice, and Eli5 were employed for model explainability, reinforcing the Random Forest classifier as the most reliable choice for this classification task.

1) Performance Metrics - Confusion Matrix: In our work, we utilize the confusion matrix and the corresponding scores to quantify the performance of different classifiers. A confusion matrix summarizes prediction results for a classification problem by breaking down the correct and incorrect predictions into count values for each class. As shown in Fig. 6, the columns represent the actual values of the categories, while the rows indicate the predicted values.

- **True Eligible/True Positive (TE)**: Eligible devices that can become UE-VBS and are correctly classified as 'eligible'.
- **False Eligible/False Positive (FE)**: Ineligible devices that cannot become UE-VBS and are incorrectly classified as 'eligible'.
- **True Ineligible/True Negative (TI)**: Ineligible devices that cannot become UE-VBS and are correctly classified as 'ineligible'.
- **False Ineligible/False Negative (FI)**: Eligible devices that can become UE-VBS and are incorrectly classified as 'ineligible'.

From the confusion matrix, several scores can be computed to evaluate the performance of different classifiers:

1) **Accuracy**, the most intuitive metric, describes the number of correct predictions out of the total predictions: Accuracy $= \frac{TE+TI}{TE+TI+FE+FI}$.
2) **Precision** (or Specificity) measures how many of the positive (eligible) predictions made are correct (true), considering both true and false positive cases: Precision $= \frac{TE}{TE+FE}$.
3) **Recall** (or Sensitivity or True Eligible Rate, TER) indicates how many of the positive (eligible) cases the classifier correctly predicted: Recall $= \frac{TE}{TE+FI}$.
4) **F1-Score** combines both precision and recall to describe the harmonic mean of the two: F1-Score $= \frac{2\times\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}}$.
5) **False Ineligible Rate (FIR)** indicates the proportion of the negative (ineligible) class that was incorrectly classified by the classifier: FIR $= \frac{FI}{TI+FI} = 1 -$ Precision.

TABLE XI: Results from the different classification models before SMOTE.

Algorithm	Precision		Recall		F1-Score		Accuracy
	0	1	0	1	0	1	
AdaBoost	0.95	0.93	0.99	0.79	0.97	0.85	0.95
Gradient Boosting	0.96	0.93	0.99	0.81	0.97	0.86	0.95
Random Forest	0.96	0.92	0.98	0.80	0.97	0.86	0.97

2) Classification Results with/without SMOTE: As discussed in Section IV-A3, SMOTE improves the performance of the classification models. The recall score of class 1 sees a significant increase for the different classifiers, as shown in the results recorded in Table XII, compared to the results reported in Table XI, obtained before applying SMOTE. Both the Gradient Boosting classifier and the Random Forest classifier achieve the same accuracy of 0.95. Furthermore, from Table XII, we see that the Random Forest classifier achieves an F1-Score of 0.88 for class 1 ('eligible' UEs), and an area

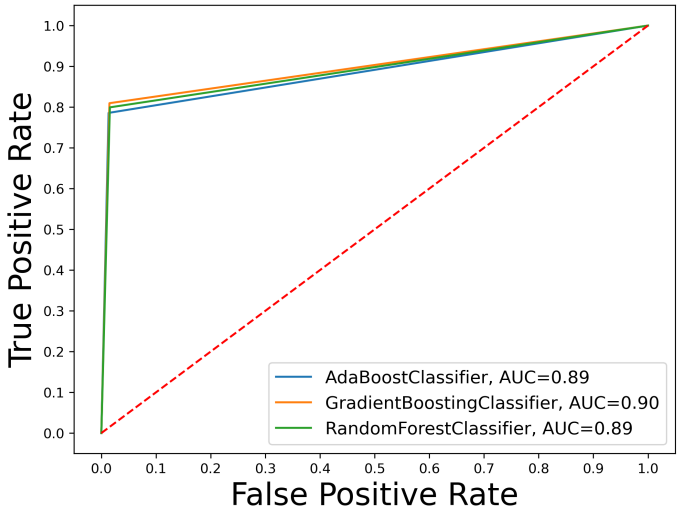


Fig. 8: Area under ROC curve for the different classification models before SMOTE (ROC AUC Analysis).

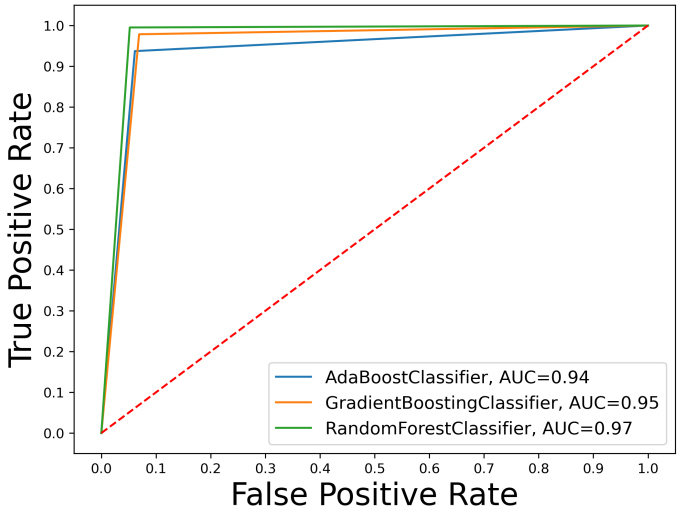


Fig. 9: Area under ROC curve for the different classification models after SMOTE (ROC AUC Analysis).

under ROC of 0.94, as demonstrated in Fig. 9. Overall, our investigations show that the Random Forest approach outperforms the rest of the models regarding F1-Score and ROC values.

TABLE XII: Results from the different classification models after SMOTE.

Algorithm	Precision		Recall		F1-Score		Accuracy
	0	1	0	1	0	1	
AdaBoost	0.98	0.82	0.95	0.90	0.97	0.86	0.94
Gradient Boosting	0.98	0.83	0.96	0.94	0.97	0.88	0.95
Random Forest	0.98	0.85	0.96	0.91	0.97	0.88	0.97

Our approach effectively handles dataset imbalance, resulting in a robust model that performs well on both majority and minority classes. As shown in Table XIII, the classification report provides insights into the model's precision, recall, and F1-Score.

	Precision	Recall	F1-Score	Support
False	0.995114	0.948758	0.971383	1288.00000
True	0.949924	0.995231	0.972050	1258.00000
Accuracy	0.971720	0.971720	0.971720	0.97172
Macro Avg	0.972519	0.971994	0.971716	2546.00000
Weighted Avg	0.972785	0.971720	0.971712	2546.00000

TABLE XIII: Classification Report

The classification report in Table XIII shows that our model achieves high precision, recall, and F1-scores for both classes, indicating that it performs well in distinguishing between the minority and majority classes. The accuracy of the model is 0.971720, which demonstrates its robustness and reliability. These results demonstrate that our approach effectively addresses dataset imbalance, ensuring that the model remains accurate and reliable across different classes.

3) ROC Curve Evaluation: The area under the curve (AUC) - receiver operating characteristics (ROC) plot (AUC - ROC) is another metric used to assess classification model performance. Table XIV presents the AUC values and their corresponding interpretations. The AUC values are calculated using the trapezoidal approximation in the following equation: $Y = [0; TER; 1]; X = [0; FIR; 1]; AUC = \text{trapz}(Y, X)$. Thus, from the table, we can discern that the higher the area under the curve, the better the model's performance. Fig. 8 shows the AUC-ROC analysis for the different classification models trained on our dataset. Notably, from Fig. 8, we can discern that the Gradient Boosting approach performs marginally better than Random Forest and AdaBoost in terms of AUC value, and hence, it can better distinguish between positive and negative class points. Furthermore, in Table XI, we report the scores obtained from the confusion matrix for the various classification models investigated in this work. The table shows that AdaBoost, Gradient Boosting, and Random Forest yield the same accuracy score of 0.95.

4) Validation, Testing & Model Explainability: This section provides insights and statistical reasoning behind selecting the machine learning model for our pipeline and the model's explainability. Model explainability is a broad concept of analyzing and understanding the results provided by ML models. It is most often used in the context of "black-box" models, for which it is difficult to demonstrate how the model arrived at a specific decision and whether the decision made is technically sound. We tested various classification algorithms for model selection with K-Fold, Hold-out Testing, and Nested Cross-validation tests, followed by SHAP, LIME, QLattice, and Eli5 tests to understand the model's explainability better.

a) K-Folds Validation & Testing: Starting with K-folds testing, it is a cross-validation test [57]. Cross-validation is

TABLE XIV: Interpreting the area under the Receiver Operator Characteristics curve.

AUC Value	Inference
$AUC = 0$	The classifier predicts all negatives as positives and all positives as negatives.
$AUC = 0.5$	The classifier is unable to distinguish positive and negative class points, predicting a random or constant class for all data points.
$0.5 < AUC < 1$	There is a high chance that the classifier will be able to distinguish between the two classes.
$AUC = 1$	The classifier can perfectly distinguish between all positive and negative class points correctly.

a resampling procedure used to evaluate machine learning models on a limited data sample. The procedure has a single parameter called K, which refers to the number of groups into which a given data sample is to be split (as shown in Algorithm 4).

Algorithm 4 Cross-Validation Algorithm

1: Shuffle the dataset randomly.
2: Split the dataset into K groups.
3: **for** each unique group **do**
4: Take the group as the test data.
5: Take the remaining groups as training data.
6: Fit a model on the training data and evaluate it on the test data.
7: Retain the evaluation score and discard the model.
8: **end for**
9: Summarize the model's skill using the model evaluation scores as the sample.

We tested several algorithms, and the results were visualized as a box plot, as shown in Fig. 10. Algorithms with higher mean and lower variance of the accuracy score are considered efficient models. Our test outcomes are documented in Table XV. From the test results presented in Table XV, it is evident that the Random Forest classifier is the ideal candidate algorithm for our pipeline as it results in the highest mean accuracy score of 0.97.

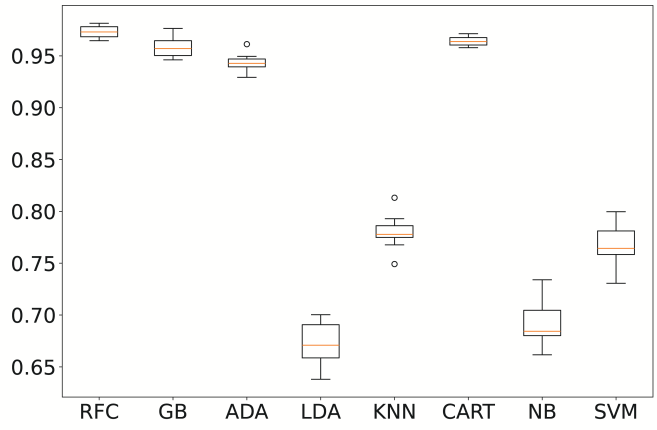


Fig. 10: K-fold cross-validation with k=10 for various classification algorithms with accuracy as the performance metric.

TABLE XV: K-fold testing using various classifiers.

Classification Algorithm	Mean	Variance
Random Forest	0.97	0.023
Gradient Boosting	0.95	0.024
AdaBoost	0.93	0.014
K-Nearest Neighbours	0.82	0.036
Naive Bayes	0.69	0.211
Support Vector Machines	0.69	0.229
Linear Discriminant Analysis	0.62	0.047

b) Nested cross-validation Validation and Hold-out Testing: To expand our testing and validation of the model, we examined it using hold-out testing and nested cross-validation. The model demonstrated impressive performance on the hold-out testing set with an accuracy of 0.981. According to the classification report in Table XVI, the precision for the False class was 0.993, the recall was 0.983, and the F1-score was 0.988, based on support of 854 instances. For the True class, the precision was 0.931, the recall was 0.969, and the F1-score was 0.949, with a support of 196 instances. Overall, the accuracy of the model was 0.981 for the 1050 instances in the testing set. The macro average precision, recall, and F1-score were 0.962, 0.976, and 0.968, respectively, while the weighted average precision, recall, and F1-score were 0.976, 0.970, and 0.972. Moreover, the nested cross-validation process yielded an accuracy of 0.9786 with a standard deviation of 0.0045, as shown in Table XVII. The best hyperparameters found through GridSearchCV are detailed in Table XVIII, indicating that the optimal model configuration includes bootstrap=True, max_depth=20, min_samples_leaf=1, min_samples_split=2, and n_estimators=100. These results underscore the robustness and reliability of the model.

TABLE XVI: Classification Report

Class	Precision	Recall	F1-Score	Support
False	0.993	0.983	0.988	854
True	0.931	0.969	0.949	196
Accuracy	0.981 (1050 instances)			
Macro Avg	0.962	0.976	0.968	1050
Weighted Avg	0.976	0.970	0.972	1050

c) Model Explainability: SHAP and LIME [40] values are used to explain the interpretability of machine learning models. SHAP and LIME are model-agnostic, meaning they are model-independent. These values are measured by

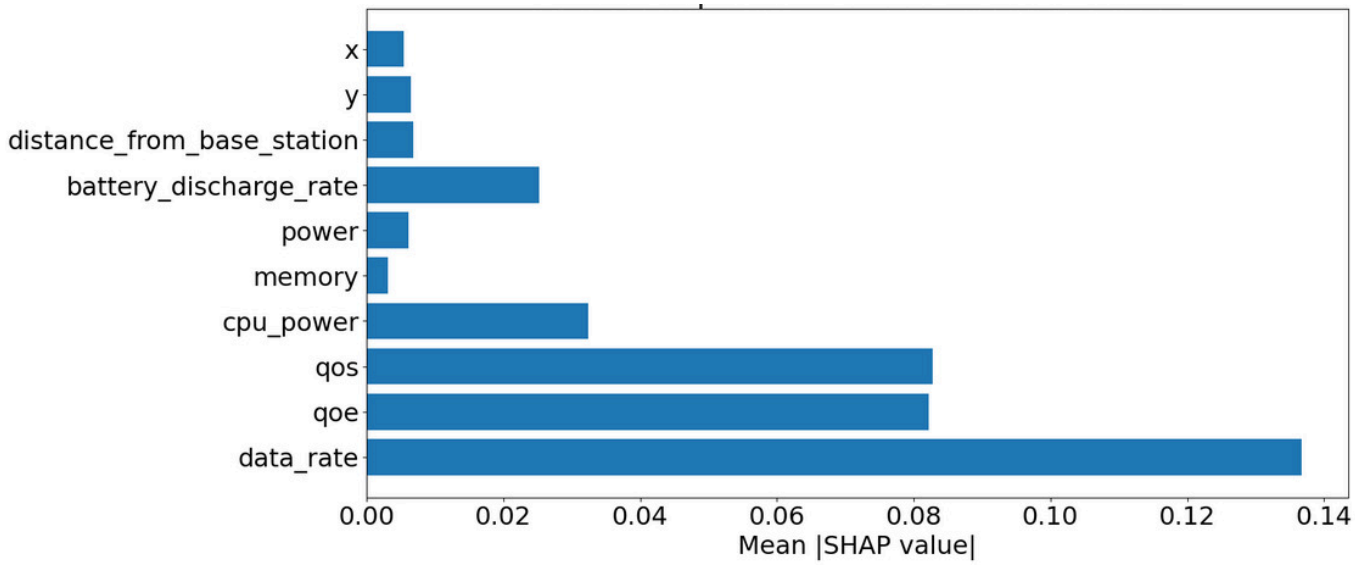


Fig. 11: SHAP Examination of Feature Importance.

TABLE XVII: Nested Cross-Validation Accuracy

Metric	Value
Accuracy	0.9786
Standard Deviation	0.0045

TABLE XVIII: Best Hyperparameters

Hyperparameter	Value
bootstrap	True
max_depth	20
min_samples_leaf	1
min_samples_split	2
n_estimators	100

calculating the impact of various parameters. LIME values are calculated by local sub-sampling of the dataset, whereas SHAP values are calculated by removing certain features and calculating their importance.

LIME attempts to approximate the mapping function $f(x)$ of the machine learning model by sampling instances (referred to as input perturbation). LIME generates a set of synthetic samples x', closely based on the original instance x. LIME then passes x' to the original model f and records the respective prediction. This process enables LIME to determine how the different input fluctuations influence f. At the end of the process, for a given sample x, LIME could approximate the prediction of f by determining the individual influence of every feature. Therefore, LIME can explain a specific prediction by understanding which features contribute the most to the prediction. Figs. 11 and 12 illustrate the SHAP and LIME values, respectively.

In this work, SHAP values calculate the importance of a feature by comparing what a model predicts with and without the feature. Since the order in which a model sees features can affect its predictions, this is done in every possible order to compare the features fairly. Thus, in our work, all possible coalitions (sets) of feature values are evaluated with and without a selected feature to calculate the exact SHAP value.

The LIME value explains why a UE is classified as eligible or not. Fig. 12 shows why the selected UE is eligible to act as a VBS, and the corresponding LIME scores are computed for each parameter. Here, we notice that a UE with a high data rate and high computational power influences the model's decision to choose that UE as an eligible VBS. This is because these high values of the considered parameters lead to high LIME scores.

While LIME explains the interpretation of a single prediction (local interpretation), the SHAP value explains the decisions the model makes globally. This is again in line with our earlier explanation of LIME. The parameters data rate, CPU power, and battery discharge rate have higher SHAP values and thus heavily influence the decisions of the classifier. Thus, from Figs. 11 and 12, we can see that UEs with higher data rates and CPU power are likely to belong to class 0 (eligible) and UEs with low memory and very high battery discharge rates tend to belong to class 1 (not eligible). This inference, which seems very intuitive, explains the reason behind selecting a classifier, thereby giving more confidence in the decisions made by the chosen classifier.

Furthermore, in contrast to the study in [34], which is based on random selection, in our work, we have studied the performance of the models in picking up the UEs eligible to become UE-VBSs based on the network parameters.

The results of our study are reported in Table XIX. From Table XIX, it can be noticed that our model-based selection of the UEs to act as UE-VBSs achieves a five-fold increase in data rate compared to the random selection presented in [34]. Besides, the received power and QoE achieved are considerably higher when our proposed ML model-based selection is

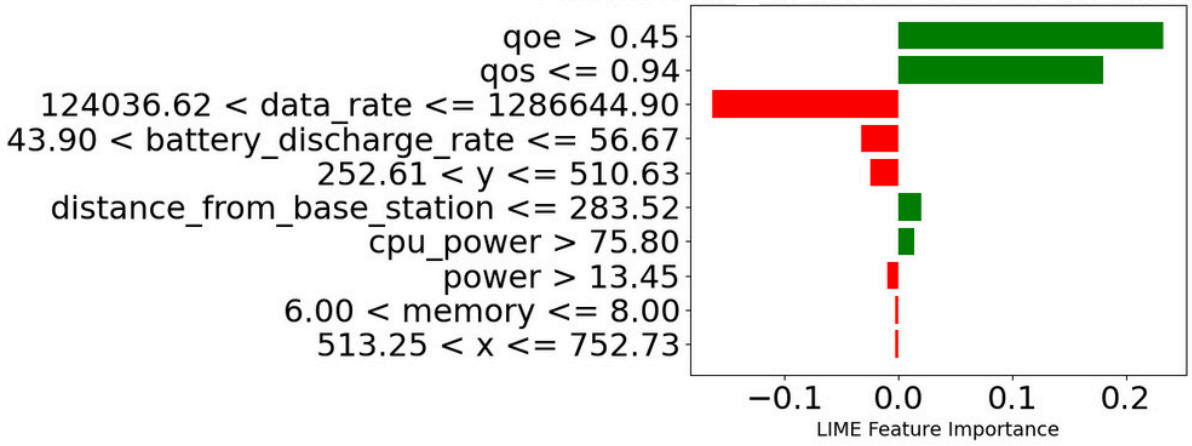


Fig. 12: LIME Examination of Features Importance.

TABLE XIX: Comparison of the Achievable Network Metrics-Random Selection and Model Selection.

Average Network Performance Metric	Random Selection	Model Selection
Data Rate (bits per second)	46.5 Mbps	247.6 Mbps
Received Power (dBm)	11.60	11.86
QoE	0.24	0.34
QoS	0.96	0.95

invoked than those attained in [34].

In addition to the methods above, the study by Palkar et al. (2024) presents a comprehensive approach to glioma prognosis using machine learning models enhanced with Explainable AI (XAI) techniques. The method provides detailed insights into the model's decision-making process and the relative importance of various features in the classification task. The approach integrates several XAI methods, including SHAP (SHapley Additive exPlanations), LIME (Local Interpretable Model-agnostic Explanations), QLattice, and Eli5, to interpret and elucidate the predictions made by machine learning models such as Support Vector Machines, Random Forests, Logistic Regression, and Decision Trees. This multifaceted interpretative strategy enables a transparent evaluation of the molecular and clinical features influencing glioma prognosis, empowering medical professionals with actionable insights for improved diagnosis and treatment planning [47]. Thus, we also examine our model with QLattice and Eli5 in our examination.

Table XX presents the feature importance results derived from the Eli5 method. The table highlights the relative importance of each feature used in the model. The 'data_rate' feature shows the highest importance with a value of 0.361013, indicating its significant influence on the model's predictions. The features 'QoE' and 'QoS' follow with importance values of 0.157479 and 0.126620, respectively. Other features, such as 'cpu_power' and 'battery_discharge_rate', also contribute to the model but to a lesser extent. Interestingly, some features like 'distance_from_base_station', 'x', and 'power' have negative importance values, suggesting they may not be beneficial for the model or could introduce noise. Overall, the results provide a clear hierarchy of feature importance, which is crucial for understanding the model's decision-making process and improving its accuracy.

TABLE XX: Eli5 Feature Importance Results

Feature	Importance
data_rate	0.361013
QoE	0.157479
QoS	0.126620
cpu_power	0.062898
battery_discharge_rate	0.050883
y	0.000824
memory	0.000824
distance_from_base_station	-0.000118
x	-0.000589
power	-0.000942

Fig. 13 shows the results from the QLattice model. The ROC curves and confusion matrices for both the training and test sets are presented. The model achieves an AUC of 0.89 for both sets, indicating good performance. The confusion matrices further break down the model's predictions, showing a high number of true positives and true negatives, which suggests the model effectively distinguishes between the classes. The combination of these results provides a comprehensive evaluation of our model. The Eli5 feature importance highlights the most influential features, while the QLattice results demonstrate the model's predictive performance. Together, these insights offer a transparent and interpretable understanding of the model's behavior and decision-making process.

E. Comparative Evaluation Results of Framework

In this section, we comprehensively evaluate our proposed Two-Stage ML Heuristic, comparing its performance against various state-of-the-art approaches for both sum rate enhancement and power consumption in 5G and beyond networks. The

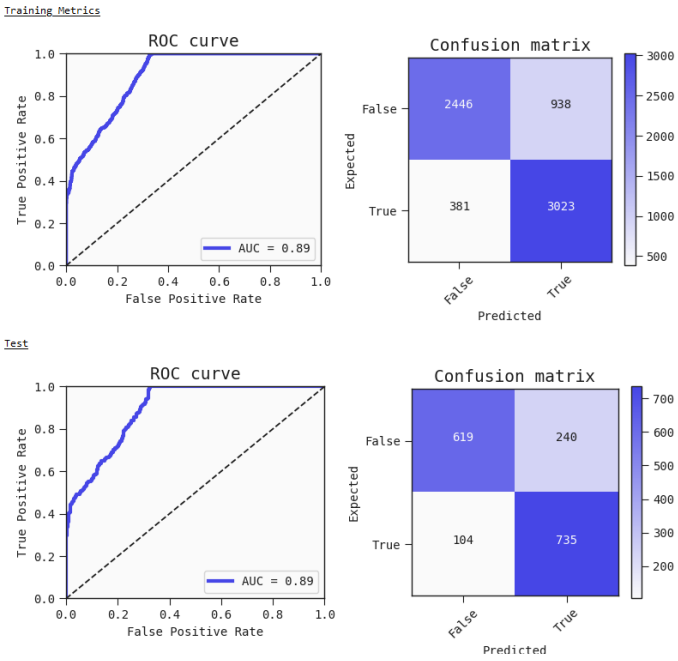


Fig. 13: QLattice Model Results: ROC curves and confusion matrices for training and test sets

approaches under comparison include Affinity Propagation [56], Grid-based Clustering [14], and a Random selection approach. Our analysis covers a range of scenarios with varying numbers of devices, from 100 to 1000, enabling a thorough investigation of the system's scalability and energy efficiency. For our sum rate a and power consumption experiments, we utilized the Telecom Italia and OPNET datasets, which are publicly available at IEEE Dataport[6]. These datasets provide realistic network traffic data, enhancing the validity and practical applicability of our findings.

1) Sum Rate: For our sum rate experiment, the number of UEs (User Equipment) varied from 100 to 1000, increasing in increments of 100. In each experiment, UEs were randomly positioned within a 1000x1000 meter area. At each stage, UEs were clustered using the MeanShift clustering algorithm, and eligible UEs within each cluster were selected as VBS (Virtual Base Stations) based on proximity to the cluster center using our custom heuristic. Specifically, UEs with the shortest distance to the cluster center were selected as VBS. In parallel, UEs were also clustered using the Affinity Propagation method, and VBS selection was performed for comparison. To assess the system's scalability, we measured the computational cost for different UE densities. Our heuristic algorithm demonstrated a time complexity of $O(n)$ for VBS selection, ensuring rapid execution even in large-scale scenarios. However, clustering complexity varied based on the algorithm: MeanShift required $O(n^2t)$, whereas Affinity Propagation exhibited similar quadratic complexity, impacting performance at higher UE densities. The most computationally efficient method, BIRCH ($O(n)$ complexity), demonstrated superior scalability, suggesting its suitability for ultra-dense scenarios. These findings highlight the need for selecting computationally efficient clustering techniques when operating in dense network environments.

As illustrated in Figure 14, our Two-Stage ML Heuristic consistently outperforms the other methods across the entire range of device counts. The heuristic demonstrates superior performance, particularly as the number of devices increases, with sum rates significantly higher than those achieved by the Affinity Propagation [56] and Grid-based Clustering [14] methods. The performance gap becomes more pronounced as the number of devices increases, with the heuristic maintaining a strong upward trend in the sum rate, ultimately reaching nearly 1.8 billion bps at 1000 devices. This superior performance is attributed to the heuristic's ability to intelligently select UEs as virtual base stations (VBSs) by leveraging real-time data rates and SINR metrics. The Grid-based Clustering method [14] also shows competitive performance, particularly in scenarios with a moderate number of devices (200-600 devices), where it closely matches the sum rate of our heuristic. However, as the number of devices approaches 1000, the Grid-based method begins to plateau while our heuristic continues to scale effectively, further widening the performance gap. In contrast, the Affinity Propagation method [56], although showing improvements over the Random selection approach, remains less effective in maximizing the sum rate, particularly in high-density scenarios. The method's sum rate peaks at a lower level and exhibits more fluctuations, indicating less consistency in performance. The impact of large-scale deployment is particularly noticeable when evaluating the computational efficiency of clustering methods. As UE densities increase beyond 500, approaches with $O(n^2)$ complexity (such as Affinity Propagation and Mean-Shift) experience noticeable slowdowns, whereas algorithms like BIRCH, which operate in $O(n)$ time, maintain consistent performance.

To further investigate the impact of increasing UE densities, we analyzed the trade-offs between computational complexity, clustering efficiency, and sum rate gains within our **two-stage approach** (Stage 1: SOM-based clustering; Stage 2: Random Forest classification and VBS selection):

- **Computational Complexity (Stage 1: SOM Clustering)**: In our framework, SOM are employed in the first stage of clustering. While SOM preserves topological relationships and often yields higher-quality clusters, it typically operates in $O(n^2)$ time, which can be challenging in ultra-dense networks (e.g., 1000+ UEs). By contrast, BIRCH, with $O(n)$ complexity, scales more easily but may compromise cluster fidelity.
- **Sum Rate Performance (Stages 1 + 2)**: Despite the heavier computational load of SOM, our **two-stage ML heuristic** achieves the highest overall sum rate. After SOM-based clustering, we invoke the second stage—a Random Forest classifier—to identify and activate the

[6]Telecom Italia and OPNET Datasets for Network Traffic Prediction. Available: https://ieee-dataport.org/documents/telecom-italia-and-opnet-datasets-network-traffic-prediction\#files

most capable UEs as VBSs. By leveraging real-time data rates, SINR, battery, and CPU metrics within this classification, the network benefits from efficient traffic aggregation, reduced interference, and superior throughput.

- **Trade-Offs and Optimization**: In real-world deployments, the choice of clustering (Stage 1) must balance accuracy and scalability. SOM excels at moderate UE densities, thanks to its ability to preserve topological features, but may become computationally expensive at very large n. Conversely, more scalable methods (e.g., BIRCH) can handle extreme densities but risk lower cluster accuracy. Meanwhile, the Random Forest classifier (Stage 2) ensures that only high-capability UEs become VBSs, further enhancing the sum rate and optimizing power usage. A *hybrid approach*—using SOM at moderate densities and switching to BIRCH at higher densities—could unify the strengths of both clustering strategies under the same **two-stage** pipeline, maximizing throughput without incurring prohibitive runtime costs.

The comparison demonstrates that our Two-Stage ML Heuristic outperforms recent advancements in the field and provides a more scalable and robust solution for maintaining high-quality service in dense network environments. This is particularly evident as the number of devices increases from 100 to 1000, where our heuristic consistently achieves the highest sum rates. The detailed comparison and analysis, as presented in Figure 14, validate the efficacy of our approach in real-world network conditions, making it a superior choice for optimizing network throughput in 5G and beyond networks.

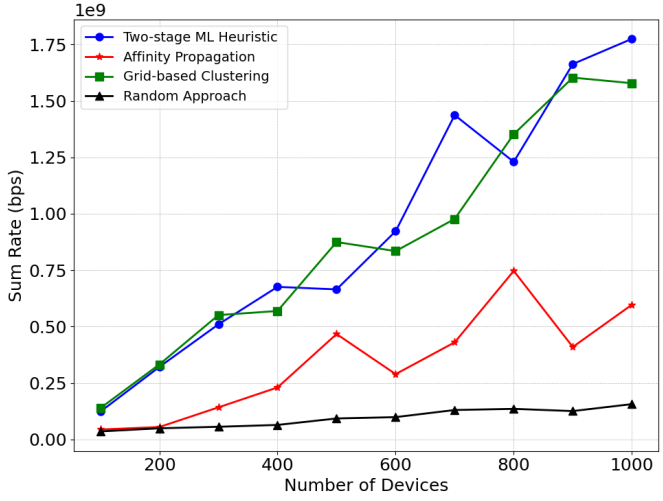


Fig. 14: Sum Rate vs. Number of Devices for Different Methods.

2) Power Consumption: To conduct the power consumption analysis, the number of UEs varies from 100 to 1000 in increments of 100, with UEs positioned randomly within a 1000x1000 meter area (consistent with the sum rate evaluation). For each scenario, we measure the power consumption of UE-VBSs based on transmission power, path loss, and interference levels. The clustering and heuristic selection processes are designed to optimize UE-VBS assignment and reduce redundant transmissions. In this study, MeanShift clustering is employed for initial VBS selection, while Affinity Propagation is used for comparison; a random selection method serves as the baseline. Our proposed heuristic algorithm, which maintains a time complexity of $O(n)$ for VBS selection, demonstrates an adaptive power-saving mechanism by intelligently optimizing spatial frequency reuse and mitigating interference. In contrast, clustering complexities vary: MeanShift requires $O(n^2t)$ operations, and Affinity Propagation exhibits similar quadratic complexity, leading to increased power consumption due to prolonged processing time. BIRCH, with its linear complexity ($O(n)$), shows higher efficiency in large-scale scenarios.

Figure 15 illustrates the total power consumption across different UE densities. As the number of devices increases, the proposed heuristic consistently outperforms Affinity Propagation and Grid-based Clustering [14]—the latter being competitive only at moderate densities (200–600 devices) but less efficient at extreme densities (800–1000 devices). Meanwhile, the random selection baseline consumes the highest amount of power due to the absence of an intelligent selection mechanism.

To further analyze power consumption trends under large-scale deployment, we examined key performance trade-offs across different clustering (Stage 1) and selection (Stage 2) strategies:

- **Rate of Increase**: Power consumption generally rises with UE density. Nonetheless, our **two-stage pipeline** (SOM for clustering in Stage 1, followed by Random Forest selection in Stage 2) maintains a lower rate of power increase compared to single-stage or random-selection methods.
- **Clustering Complexity (Stage 1)**: Quadratic-complexity algorithms such as MeanShift and Affinity Propagation experience higher energy costs at extreme UE densities due to prolonged computations. By contrast, BIRCH ($O(n)$ complexity) remains more scalable, consuming less power under ultra-dense conditions. While our approach employs SOM for improved cluster accuracy, we note that BIRCH is a strong alternative where power efficiency under large n is paramount.
- **Energy-Efficient Allocation (Stage 2)**: In the second stage, our Random Forest classifier identifies the most suitable UEs as VBSs, ensuring an energy-efficient resource allocation. By activating only high-capability devices with favorable SINR and battery levels, we reduce interference and optimize overall power distribution.
- **Hybrid Clustering Possibilities**: Although we primarily use SOM in Stage 1, a hybrid approach—employing MeanShift for small-to-moderate densities and switching to BIRCH at higher densities— can further optimize power efficiency. This adaptive strategy balances the cluster quality of SOM or MeanShift for fewer UEs

with the superior scalability of BIRCH when the network grows ultra-dense.

These results highlight the importance of computationally efficient clustering methods in power-sensitive applications. The Two-Stage ML Heuristic not only maximizes the sum rate but also significantly reduces power consumption, making it a practical solution for large-scale 5G and beyond networks.

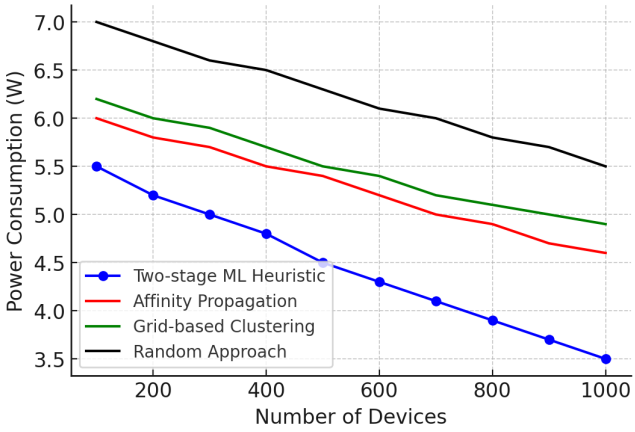


Fig. 15: Power Consumption vs. Number of Devices for Different Methods.

3) Real-Time of Response/Execution Time: In addition to power consumption, it is crucial to evaluate the real-time performance and execution time of our Two-Stage ML Heuristic relative to alternative methods to ensure the framework's viability in practical network deployments where latency and computational overhead are critical. In our experiments, we vary the number of UEs from 100 to 1000 in increments of 100 and measure the total runtime required to (i) cluster the devices, (ii) classify eligible UE-VBS nodes, and (iii) apply the heuristic for final VBS selection. While our approach is highly accurate and robust, it may incur greater computational costs than simpler solutions due to the added complexity of SOM-based clustering and ensemble classification.

Figure 16 illustrates the response time versus the number of UEs for our Two-Stage ML Heuristic, Affinity Propagation, Grid-based Clustering, and a random selection baseline. An increasing trend in response time is observed for all methods as the network size grows; however, the steeper increase for our heuristic—attributable to the potential $O(n^2)$ overhead of SOM clustering and the computational demands of Random Forest classification—highlights the trade-off between model sophistication and operational scalability in real-world networks.

As the figure indicates, our approach requires more processing time than more straightforward methods, particularly at large UE densities where computationally expensive clustering can dominate overall runtime. Despite this overhead, the significant gains in network throughput, power savings, and coverage justify the complexity in many use cases, mainly where high accuracy and resilience are paramount.

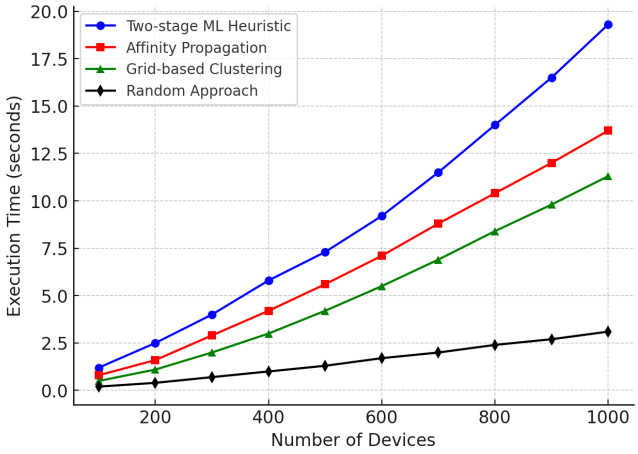


Fig. 16: Response Time vs. Number of Devices for Different Methods.

VI. Conclusions And Future Work

This paper investigates and implements a Two-Stage ML Heuristic framework for 5G/6G mobile network augmentation through dynamic selection and activation of UE-VBSs. We generated and pre-processed the dataset by simulating the UEs' geographical coordinates as a non-uniform distribution in both the time and space domains. Additionally, another dataset was generated to classify the devices in each cluster into eligible UE-VBS and ineligible UE-VBS categories. Using these datasets, we trained a two-stage machine learning model, followed by a heuristic algorithm for selecting one eligible device per cluster to become a UE-VBS.

When employing SOM for clustering, we observed the SOM's ability to preserve the topological and neighborhood relationships among UEs, enabling more natural grouping in a high-dimensional setting. Alternative techniques such as K-means or DBSCAN typically require explicit assumptions on cluster shape or number and do not inherently retain topological ordering, often leading to less robust separations for subsequent classification. Likewise, for classification, Random Forest provided a balance of interpretability, robustness against overfitting, and ease of handling heterogeneous features. While methods such as SVM[7] or deep neural networks can yield high performance, they often demand extensive hyperparameter tuning and can obscure decision boundaries. Thus, the specific combination of SOM and Random Forest was selected due to their complementary strengths of interpretability, computational feasibility, and consistent performance observed in our experiments.

[7]Although a basic linear SVM can appear simple, practical scenarios often require non-linear kernels (e.g., RBF or polynomial), each with multiple parameters (e.g., C, γ), plus sensitivity to feature scaling. This increases the computational load and complicates tuning, particularly for large-scale 5G/6G data. In contrast, ensembles like Random Forest tend to have more robust defaults and simpler interpretability, making them better suited to our dual-stage pipeline.

Our analysis shows that the best performance was achieved using SOM for clustering combined with the Random Forest classifier for device eligibility classification. SOM consistently outperformed other clustering techniques due to its ability to preserve topological relationships in non-linear and large-scale datasets, attaining an improved silhouette score of 0.64. In contrast, other clustering methods—such as Affinity Propagation, Grid-based Clustering, and even the deep learning–based ClusterGAN (which obtained a silhouette score of 0.45)—were less effective. The proposed model was also subjected to rigorous statistical testing using SHAP and LIME, which further validated its effectiveness. From the performance analysis, we advocate that our ML-aided dynamic selection of a UE-VBS for each cluster around the primary BS is highly efficient, having consistently resulted in a significantly higher sum rate compared to random selection and outperforming alternative clustering approaches. Paired with the Random Forest classifier—which achieved high accuracy and recall in classifying eligible UE-VBS devices—this combination proves superior to the other tested approaches.

More specifically, the comparative evaluation results demonstrate that our Two-Stage ML Heuristic consistently outperforms state-of-the-art clustering methods in terms of sum rate, power consumption, and scalability. Across all device densities, our approach achieved the highest sum rate, with a peak of nearly 1.8 billion bps at 1000 UEs, outperforming Affinity Propagation and Grid-based Clustering. Additionally, the heuristic significantly reduced power consumption by optimizing UE-VBS selection, effectively minimizing redundant transmissions and interference, making it a more energy-efficient solution for large-scale 5G networks. While our approach required a higher computational cost due to the complexity of SOM clustering and Random Forest classification, the performance benefits in network throughput, energy efficiency, and scalability justify the additional processing overhead, making it a robust and practical solution for real-world deployments.

Furthermore, while the proposed approach demonstrates significant benefits in terms of data rate gains and selection accuracy, future real-world deployments will necessitate an in-depth understanding of trade-offs among accuracy, power consumption, and computational complexity. For instance, clustering with SOM may outperform simpler methods (e.g., BIRCH) in terms of cluster quality but also demands higher computational resources as network density grows. Similarly, ensemble-based classifiers like Random Forest offer robust performance yet can incur more training overhead than single-tree models. In large-scale or resource-constrained scenarios, choosing a more lightweight algorithm might reduce power usage at the cost of slightly lower accuracy. Therefore, the final system design must carefully balance these performance metrics—potentially via adaptive or hybrid methods—to ensure efficient, scalable, and cost-effective network operation across diverse environments.

Finally, while our proposed solution demonstrates strong results in terms of both network throughput and classification accuracy, it is important to acknowledge the inherent trade-offs among accuracy, power consumption, and computational complexity. For instance, resorting to a more sophisticated clustering method—such as SOM—can yield higher clustering fidelity but at the expense of increased computational overhead, which may not be ideal in large-scale or resource-constrained environments. Likewise, adopting ensemble techniques like Random Forest typically boosts model accuracy and resilience against noise but demands more energy and CPU time compared to simpler algorithms. Balancing these factors is essential for real-world applicability: in scenarios where power efficiency is paramount (e.g., battery-limited IoT devices), a slightly less accurate yet computationally lighter model might be more appropriate. Conversely, high-stakes applications—such as mission-critical communications—may tolerate higher resource usage if it translates into significantly improved reliability. Going forward, a systematic analysis of these multidimensional trade-offs would help tailor the solution more precisely to different deployment scenarios, ensuring both scalability and performance efficiency.

In future work, we will explore hybrid or adaptive approaches (e.g., switching from SOM to a more lightweight algorithm at high densities) to reduce execution time while retaining acceptable accuracy and robustness. Additionally, real-world validation is crucial to prove the UE-VBS framework's robustness and scalability since diverse environments, interference, and legacy devices cannot be fully replicated in simulations. We plan to collect and utilize real datasets from small-scale testbeds to further refine clustering algorithms, UE-VBS selection, and data-driven methods, then expand into larger trials with network operators to evaluate throughput, latency, and reliability under realistic conditions. Security threats like eavesdropping, spoofing, and denial-of-service will be analyzed, alongside user consent, data privacy, and regulatory compliance. Non-uniform user densities will drive the exploration of advanced clustering algorithms and machine-learning techniques informed by real-time channel indicators. A comprehensive security framework will mitigate vulnerabilities, while integration with IoT, edge computing, and dynamic spectrum management will enhance resource utilization and sustainability. Regulatory and policy discussions will address spectrum allocation, user consent, and device compensation. Ultimately, UE-VBS offers a flexible, cost-effective way to meet rising network demands, paving the way for resilient, high-capacity mobile networks ready for 5G and beyond.

VII. Acknowledge

This work is implemented with the contribution of Nivedhitha D and Sriram M. Moreover, it has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No. 739578, the ADROIT6G project of the SNS-JU under Grant Agreement No. 101095363, and the Government of the Republic of Cyprus through the Deputy Ministry of Research, Innovation and Digital Policy.

APPENDIX

A. Background Work Related to AI/ML

ML has significantly advanced automation, reducing the need for manual labor and control. In many instances, ML outperforms traditional statistical methods in terms of efficiency and reliability, revolutionizing various research areas and having a widespread impact on the digital world. Numerous innovative techniques have been developed for artificial intelligence (AI) and ML tasks. Broadly, ML algorithms are categorized into two types: unsupervised learning and supervised learning. The following sections will explore clustering techniques associated with unsupervised learning and classification techniques related to supervised learning.

Unsupervised learning [23] discovers patterns in unlabeled datasets, identifying structures and forming groups within data. Clustering partitions data points into groups where points in the same group are more similar than those in different groups.

Supervised learning [16] employs algorithms on pre-labeled data, consisting of predefined inputs (features) and outputs (labels). If outputs are continuous, the problem is regression; if discrete, it is classification [27]. Classification algorithms use training data to categorize new observations into predefined classes (e.g., Yes/No, 0/1), also known as targets or labels.

1) Background Work Related to ML Clustering (Unsupervised Methods):

a) K-Means Clustering Algorithm (Unsupervised): The K-means algorithm [3] clusters UE based on location using Euclidean distance and a predefined number of clusters K. It initializes K centroids randomly, assigns each data point to the nearest centroid, and updates centroids by averaging assigned points. This process repeats until centroids stabilize. The optimal number of clusters $\tilde{k}$ is often determined using the Elbow method by plotting the sum of squared errors (SSE) against increasing $\tilde{k}$ and selecting the point where SSE sharply decreases.

b) Mean-Shift Algorithm (Unsupervised): Mean-shift [10] iteratively shifts data points toward the highest density regions (modes) without requiring the number of clusters beforehand. It operates with kernel density estimation (KDE), where the density function is defined as $f_k(u) = \frac{1}{\hat{n}\bar{h}^d}\sum_{i=1}^{\hat{n}} \bar{K}\big(\frac{u-u_i}{\bar{h}}\big)$, ensuring $\int \bar{K}(u)\,du = 1$ and $\bar{K}(u) = \bar{K}(|u|)$. Although effective, it has a computational complexity of $O(n^2)$.

c) Spectral Clustering (Unsupervised): Spectral clustering [45] transforms clustering into a graph-partitioning problem by constructing a similarity graph with an adjacency matrix $\mathcal{A}$ and a degree matrix $\mathcal{D}$ where $\mathcal{D}_{ii} = \sum_j w_{ij}$. The graph Laplacian is defined as $\mathcal{L} = \mathcal{D} - \mathcal{A}$. After normalizing $\mathcal{L}$, eigenvalues and eigenvectors are used for dimensionality reduction, and clustering is subsequently performed using methods like K-means.

d) Agglomerative Clustering (Unsupervised): Agglomerative clustering [49] is a hierarchical, bottom-up approach that merges clusters based on linkage criteria such as complete linkage $\max\{d(a,b) : a \in \mathbb{A}, b \in \mathbb{B}\}$ or single linkage $\min\{d(a,b) : a \in \mathbb{A}, b \in \mathbb{B}\}$, without the need to specify the number of clusters in advance.

e) Affinity Propagation (Unsupervised): Affinity Propagation [21] clusters data by passing messages between points to identify exemplars without predefining the number of clusters. It updates the responsibility $r(i,k) \leftarrow s(i,k) - \max_{k' \neq k}\{a(i,k') + s(i,k')\}$ and availability $a(i,k) \leftarrow \min\big(0, r(k,k) + \sum_{i' \notin \{i,k\}} \max(0, r(i',k))\big)$ iteratively until convergence.

f) BIRCH (Unsupervised): BIRCH [62] efficiently clusters large datasets by creating a compact summary using clustering features (CF) defined as $CF = (\mathbb{N}, \overrightarrow{\mathbb{LS}}, \mathbb{SS})$, where $\overrightarrow{\mathbb{LS}} = \sum_{i=1}^{\mathbb{N}} \overrightarrow{\mathbb{X}_i}$ and $\mathbb{SS} = \sum_{i=1}^{\mathbb{N}} (\overrightarrow{\mathbb{X}_i})^2$. This summary enables a linear runtime complexity of $O(n)$.

g) OPTICS (Unsupervised): OPTICS [6] is a density-based algorithm requiring parameters ε and MinPts. A point p is defined as a core point if $|\mathcal{N}_\varepsilon(p)| \geq$ MinPts. The core distance is given by

$$\text{CD}(p) = \begin{cases} \text{null} & \text{if } |\mathcal{N}_\varepsilon(p)| < \text{MinPts} \\ \min(\mathcal{N}_\varepsilon(p)) & \text{otherwise} \end{cases},$$

and the reachability distance between points p and o is $\text{RD}(o,p) = \max(\text{CD}(p), \text{dist}(p,o))$.

h) SOM Algorithm (Unsupervised): The SOM algorithm [37] projects high-dimensional data onto a lower-dimensional grid, preserving topological relationships. It initializes neuron weights randomly and iteratively assigns each input x to the Best Matching Unit (BMU) by minimizing $d(i) = \sum_{k=1}^{n}(x_k - w_{ik})^2$. The BMU and its neighbors update their weights as $w_i(t+1) = w_i(t) + \alpha(t)h_{BMU,i}(t)(x - w_i(t))$, with the learning rate $\alpha(t)$ and neighborhood function h decreasing over time.

i) Deep Learning-Based Clustering: ClusterGAN (Unsupervised): Recent advances in deep learning have yielded novel unsupervised clustering methods. For example, ClusterGAN [43] integrates a generative adversarial network (GAN) with clustering objectives in the latent space to simultaneously learn representations and cluster assignments. Although ClusterGAN is capable of capturing complex non-linear data distributions, our experiments indicate that for clustering UEs based primarily on Euclidean spatial features, SOM provides superior performance in terms of clustering quality and computational efficiency.

2) Background Work Related to ML Classification (Supervised Methods):

a) Random Forest (Supervised): Random Forest [39] is an ensemble method that builds multiple decision trees using bootstrap samples from the training set. By combining bagging (bootstrap aggregating) and feature randomness, it creates an uncorrelated forest where each tree splits data based on a randomly selected subset of features. The splitting criterion often involves measures like Gini impurity, defined as $Gini = 1 - \sum_{k=1}^{C} p_k^2$, where p_k is the proportion of class k in the node. Feature importance is then determined by the average decrease in impurity across all trees. By averaging the

predictions of diverse trees, Random Forest reduces variance and mitigates overfitting, albeit with a slight increase in bias.

b) Adaptive Boosting (AdaBoost) (Supervised): AdaBoost [54] is an ensemble technique that sequentially trains weak learners on weighted versions of the training data. Initially, all instances are assigned equal weights. After each iteration, the weights of misclassified instances are increased while those of correctly classified instances are decreased. This process focuses subsequent learners on harder cases. The weight update rule is $w_i \leftarrow w_i \exp(-\alpha_t y_i h_t(x_i))$, where α_t is the weight assigned to the weak learner h_t, y_i is the true label, and x_i is the input feature vector. Final predictions are made by a weighted majority vote of all weak learners, thereby reducing bias and variance.

c) Gradient Boosting (Supervised): Gradient Boosting [29] constructs models sequentially by optimizing a loss function using gradient descent. Each new model attempts to correct the errors (residuals) of the combined ensemble so far. The update rule for the model at iteration t is $F_t(x) = F_{t-1}(x) + \eta h_t(x)$, where η is the learning rate and $h_t(x)$ is the new weak learner trained on the residuals $r_{i,t} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x_i)=F_{t-1}(x_i)}$. By iteratively minimizing the loss function, Gradient Boosting effectively reduces prediction error, creating a strong predictive model from multiple weak learners.

d) Reinforcement Learning for Dynamic Network Optimization (Interactive/Supervised): Recent research has also explored reinforcement learning (RL) for dynamic network optimization. RL methods interact with the environment to learn optimal resource allocation or UE selection policies through trial and error. Unlike the classification methods described above, RL continuously updates its policy based on rewards obtained from the environment. Although RL is promising for dynamic, real-time optimization tasks, its complexity in reward formulation and the need for continuous feedback make supervised methods such as Random Forest, AdaBoost, and Gradient Boosting more practical for our UE-VBS selection application. Nonetheless, RL remains a compelling alternative for future work.

B. Abbreviation

Table XXI provides a comprehensive list of abbreviations used throughout this paper. Our framework leverages advanced clustering algorithms—such as SOM, K-Means, Mean-Shift, Spectral and Agglomerative Clustering, BIRCH, OPTICS, Affinity Propagation, and ClusterGAN—to group UEs, while supervised methods including RF, AdaBoost, Gradient Boosting, KNN, NB, SVM, and LDA are utilized for classification. In addition, key network metrics (e.g., SINR, QoS, QoE) and specialized terms (e.g., GAPC, MAPC, IMCA, E-UTRAN, BDIx) are also detailed in the table. This reference helps ensure clarity and consistency as these terms recur in our discussion of dynamic UE-VBS selection in 5G and beyond networks.

TABLE XXI: List of Abbreviations

Abbreviation	**Description**
5G	Fifth Generation mobile network
6G	Sixth Generation mobile network
UE	User Equipment
VBS	Virtual Base Station
UE-VBS	User Equipment as Virtual Base Station
SBS	Small Base Station
BS	Base Station
SOM	Self-Organizing Maps
RF	Random Forest classifier
SMOTE	Synthetic Minority Over-sampling Technique
QoS	Quality of Service
QoE	Quality of Experience
SINR	Signal-to-Interference-plus-Noise Ratio
D2D	Device-to-Device
PPP	Poisson Point Process
JFI	Jain's Fairness Index
CART	Classification and Regression Tree
K-Means	K-Means clustering algorithm
Mean-Shift	Mean shift clustering algorithm
Spectral Clustering	Spectral clustering algorithm
Agglomerative Clustering	Hierarchical clustering algorithm
BIRCH	Balanced Iterative Reducing and Clustering using Hierarchies
OPTICS	Ordering Points To Identify the Clustering Structure
Affinity Propagation (AP)	Affinity Propagation clustering algorithm
ClusterGAN	Clustering using Generative Adversarial Network
AdaBoost	Adaptive Boosting algorithm
Gradient Boosting (GB)	Gradient Boosting algorithm
KNN	k-Nearest Neighbors
NB	Naive Bayes classifier
SVM	Support Vector Machines
LDA	Linear Discriminant Analysis
CH	Cell Head
VSC	Virtual Small Cell
VRN	Virtual Relay Node
GAPC	Grid Assisted Affinity Propagation Clustering
MAPC	Modified Affinity Propagation Clustering
IMCA	Initializing Matching Connection Algorithm
E-UTRAN	Evolved Universal Terrestrial Radio Access Network
BDIx	Belief-Desire-Intention eXtended agents
XAI	eXplainable Artificial Intelligence
LIME	Local Interpretable Model-Agnostic Explanations
SHAP	SHapley Additive exPlanations
QLattice	Symbolic model discovery framework
Eli5	Machine learning model interpretation tool

REFERENCES

[1] 3gpp ts 32.500 v16.0.0 (2020-07), 3rd generation partnership project; technical specification group services and system aspects; telecommunication management; self-organizing networks (son); concepts and requirements (release 16), 2020.

[2] 5g applications and use cases, 2021. Digi International.

[3] A. Ahmad and L. Dey. A k-mean clustering algorithm for mixed numeric and categorical data. *Data and Knowledge Engineering*, 63(2):503–527, 2007.

[4] I.F. Akyildiz, A. Kak, E. Khorov, A. Krasilov, and A. Kureev. Arbat: A flexible network architecture for qoe-aware communications in 5g systems. *Comput. Netw.*, 147:262–279, 2018.

[5] I.F. Akyildiz, S. Nie, S.-C. Lin, and M. Chandrasekaran. 5g roadmap: 10 key enabling technologies. *Comput. Netw.*, 106:17–48, 2016.

[6] Mihael Ankerst, Markus Breunig, Hans-Peter Kriegel, and Joerg Sander. Optics: Ordering points to identify the clustering structure. In *Sigmod Record*, volume 28, pages 49–60, 1999.

[7] D. Bhanu and R. Santhosh. An energy efficient grid-based clustering algorithm using type-3 fuzzy system in wireless sensor networks. *Wireless Networks*, 2023.

[8] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[9] N. Chawla, K. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *J. Artif. Intell. Res.*, 16:321–357, 2002.

[10] Yizong Cheng. Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):790–799, 1995.

[11] C. Christophorou, A. Pitsillides, and I. Akyildiz. Celec framework for reconfigurable small cells as part of 5g ultra-dense networks. In *2017 IEEE International Conference on Communications (ICC)*, pages 1–7. IEEE, 2017.

[12] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[13] A. R. Danisya, G. Hendrantoro, and P. Handayani. Ue clustering based on grid affinity propagation for mmwave d2d in virtual small cells. In *2023 IEEE International Conference on Industrial 4.0, Artificial Intelligence, and Communications Technology (IAICT)*, pages 38–44, 2023.

[14] Achmad Rizal Danisya, Gamantyo Hendrantoro, and Puji Handayani. Ue clustering based on grid affinity propagation for mmwave d2d in virtual small cells. In *Proceedings of the IEEE International Conference on Communications (ICC)*. IEEE, 2023.

[15] Small Cell Forum Document. Small cell siting and deployment challenges in hyperdense networks, 2017. SFC192.

[16] X. Dong, Z. Yu, and W. Cao. A survey on ensemble learning. *Front. Comput. Sci.*, 14:241–258, 2020.

[17] G. Doquire and M. Verleysen. Mutual information-based feature selection for multilabel classification. *Neurocomputing*, 122:148–155, 2013.

[18] D. Dutta, D. Paul, and P. Ghosh. Analysing feature importances for diabetes prediction using machine learning. In *2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pages 924–928. IEEE, 2018.

[19] Z. Fan, X. Gu, S. Nie, and M. Chen. D2d power control based on supervised and unsupervised learning. In *2017 3rd IEEE International Conference on Computer and Communications, ICCC 2017*, pages 558–563, 2018.

[20] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the Second European Conference on Computational Learning Theory*, pages 23–37, 1997.

[21] Brendan Frey and Delbert Dueck. Clustering by passing messages between data points. *Science*, 315:972–6, 2007.

[22] J. H. Friedman. Greedy function approximation: A gradient boosting machine. In *Annals of Statistics*, volume 29, pages 1189–1232, 2001.

[23] H. Frigui. Clustering: Algorithms and applications. In *2008 1st International Workshops on Image Processing Theory, Tools and Applications, IPTA 2008*, 2008.

[24] H. Gacanin. Autonomous wireless systems with artificial intelligence, 2018.

[25] Xiaohu Ge, Jing Yang, Hamid Gharavi, and Yang Sun. Energy efficiency challenges of 5g small cell networks. *IEEE Wireless Communications*, 24(5):94–100, 2017.

[26] A. Gotsis, S. Stefanatos, and A. Alexiou. Ultradense networks: The new wireless frontier for enabling 5g access. *IEEE Vehicular Technology Magazine*, 11(2):71–78, 2016.

[27] Derek Greene, Padraig Cunningham, and Rudolf Mayer. Unsupervised learning and clustering. In *Encyclopedia of Machine Learning*, pages 10–1007/978–3–540–75171–7_3. Springer, 2008.

[28] Q. Gu, Z. Li, and J. Han. Generalized fisher score for feature selection, 2012.

[29] Friedman J. H. Stochastic gradient boosting. *Computational Statistics and Data Analysis*, 38(4):367–378, 2002.

[30] I. Ioannou, V. Vassiliou, C. Christophorou, and A. Pitsillides. Distributed artificial intelligence solution for d2d communication in 5g networks. *IEEE Systems Journal*, 14(3):4232–4241, 2020.

[31] I. Ioannou, V. Vassiliou, C. Christophorou, and A. Pitsillides. Distributed artificial intelligence solution for d2d communication in 5g networks. *IEEE Systems Journal*, 14(3):4232–4241, 2020.

[32] Iacovos Ioannou, Prabagarane Nagaradjane, Ala' Khalifeh, Christophoros Christophorou, Vasos Vassiliou, Gundepudi V. Surya Sashank, Charu Jain, and Andreas Pitsillides. Ml-aided dynamic clustering and classification of ues as vbs in d2d communication networks. In *IEEE International Conference on Wireless Communications, Signal Processing, and Networking (WiSPNET)*. IEEE, 2023.

[33] X. Jin, A. Xu, R. Bie, and P. Guo. Machine learning techniques and chi-square feature selection for cancer classification using sage gene expression profiles. In *Data Mining for Biomedical Applications: PAKDD 2006 Workshop, BioDM 2006, Singapore, April 9, 2006. Proceedings*, pages 106–115. Springer, 2006.

[34] V. Jungnickel, K. Manolakis, W. Zirwas, B. Panzner, V. Braun, M. Lossow, and T. Svensson. The role of small cells, coordinated multipoint, and massive mimo in 5g. *IEEE Communications Magazine*, 52(5):44–51, 2014.

[35] M. Kamel, W. Hamouda, and A. Youssef. Ultra-dense networks: A survey. *IEEE Communications Surveys & Tutorials*, 18(4):2522–2545, 2016.

[36] T. Kohonen. *Self-Organizing Maps*, volume 30 of *Springer Series in Information Sciences*. 2001.

[37] Teuvo Kohonen. *Self-Organizing Maps*, volume 30 of *Springer Series in Information Sciences*. Springer, 3rd edition, 2001.

[38] V. Kumar, S. Yadav, D. Sandeep, S. Dhok, R.K. Barik, and H. Dubey. 5g cellular: Concept, research work and enabling technologies. In *Advances in Data and Information Sciences*, pages 327–338. Springer, 2019.

[39] Breiman L. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[40] Scott Lundberg and Su-In Lee. A unified approach to interpreting model predictions, 2017.

[41] S. Mollahasani, A. Eroğlu, I. Demirkol, and E. Onur. Density-aware mobile networks: Opportunities and challenges. *Comput. Netw.*, page 107271, 2020.

[42] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah. Drone small cells in the clouds: Design, deployment and performance analysis. In *2015 IEEE Global Communications Conference (GLOBECOM)*, 2015.

[43] Abhishek Mukherjee et al. Clustergan: Latent space clustering in generative adversarial networks. In *Advances in Neural Information Processing Systems*, 2018.

[44] S. Mukherjee and R. Krishnan. Clustergan: Latent space clustering in generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–10, 2019.

[45] Andrew Ng, Michael Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Adv. Neural Inf. Process. Syst.*, 2002.

[46] NGMN Alliance. Small cell cost sharing. Technical report, 2016.

[47] A. Palkar, C. C. Dias, K. Chadaga, and N. Sampathila. Empowering glioma prognosis with transparent machine learning and interpretative insights using explainable ai. *IEEE Access*, 12:31697–31718, 2024.

[48] Nisha Panwar, Shantanu Sharma, and Awadhesh Kumar Singh. A survey on 5g: The next generation of mobile communication. *Physical Communication*, 18:64–84, 2016.

[49] M. Pavan and M. Pelillo. Dominant sets and pairwise clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(1):167–172, 2007.

[50] R.J. Peter. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.*, 20, 1987.

[51] D. Prabakar and V. Saminadan. Improving spectral efficiency of small cells with multi-variant clustering and interference alignment. *Journal of Computational and Theoretical Nanoscience*, 17(5):2203–2206, 2020.

[52] Anjali Rajak and Rakesh Tripathi. Fdf-hybridfs: Towards design of a failure detection framework using hybrid feature selection method for ip core networks that connect 5g core in nfv-based test environment. *Computer Standards & Interfaces*, 87:103779, 2023.

[53] P. Swain, C. Christophorou, U. Bhattacharjee, C. M. Silva, and A. Pitsillides. Selection of ue-based virtual small cell base stations using affinity propagation clustering. In *2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)*, pages 1104–1109, 2018.

[54] Hastie T., Rosset S., Zhu J., and Zou H. Multi-class adaboost. *Statistics and its Interface*, 2(3):349–360, 2009.

[55] Anurag Thantharate and Cory Beard. Adaptive6g: Adaptive resource management for network slicing architectures in current 5g and future 6g systems. *Journal of Network and Systems Management*, 31(1):9, 2022.

[56] Anusha Vaishnav, Amulya Ratna Swain, and Manas Ranjan Lenka. Improved affinity propagation clustering for d2d communication in 5g. In *2022 OITS International Conference on Information Technology (OCIT)*, pages 403–408, 2022.

[57] Gitte Vanwinckelen and Hendrik Blockeel. On estimating model accuracy with repeated cross-validation. In *BeneLearn 2012: Proceedings of the 21st Belgian-Dutch Conference on Machine Learning*, pages 39–44, 2012.

[58] K. Venkateswararao, P. Swain, C. Christophorou, and A. Pitsillides. Dynamic selection of virtual small base station in 5g ultra-dense network using initializing matching connection algorithm. In *2019 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, pages 1–6. IEEE, 2019.

[59] K. Venkateswararao, P. Swain, C. Christophorou, and A. Pitsillides. Using ue-vbs for dynamic virtual small cells deployment and backhauling in 5g ultra-dense networks. *Computer Networks*, 189:107926, 2021.

[60] K. Venkateswararao, P. Swain, S. S. Jha, I. Ioannou, and A. Pitsillides. A novel power consumption optimization framework in 5g heterogeneous networks. *Computer Networks*, 220(July 2022):109487, 2023.

[61] H. Zhang, X. Li, and Y. Wang. Research on cross-domain recommendation based on generalized matrix factorization. *Chinese Journal of Electronics*, 2021.

[62] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: an efficient data clustering method for very large databases. In *SIGMOD Rec.*, volume 25, pages 103–114, 1996.