Noname manuscript No.
(will be inserted by the editor)

A Dynamic Two-stage Machine Learning Approach for the Selection of a UE-VBS in a 5G/6G Network

S.V.Jansi Rani · Iacovos Ioannou · Prabagarane Nagaradjane · Christophoros Christophorou · Vasos Vassiliou · Nivedhitha D · Sriram M · Andreas Pitsillides

Received: date / Accepted: date

Abstract The 5G cellular network is the current generation of mobile networks, and 6G is the future generation of mobile networks. They both focus on the identification of the current and future needs, respectively, of the heterogeneous devices in the context of wireless access in a dense network scenario. This research builds on the user equipment-based virtual base station (UE-VBS) concept, which utilizes smartphones to provide base station services to other UEs in the perimeter by capitalizing the capabilities of the new UE in terms of massive connectivity and the enhanced resources such as computing, and battery-power it offers. However, in a dynamic network architecture like the 5G network and even more in the 6G mobile network, the selection of a qualified UE to become a UE-VBS is a challenging task with the introduction of newer technologies, UE hardware configurations, and network infrastructures. In this context, to automate the successful identification of a potential UE to act as a VBS is a need, a machine learning (ML) model can be employed. Hence, in this paper, we propose and explore a two-stage machine learning approach to dynamically choose the eligible UEs that will be activated on the fly as UE-VBS to support data rate expansion and improve quality of service (QoS) in locations where infrastructure is lacking, and a more agile

This work 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.

S.V.Jansi Rani, Nivedhitha D and Sriram M
Department of CSE, Sri Sivasubramaniya Nadar College of Engineering, Chennai-India

Iacovos Ioannou, Christophoros Christophorou, Vasos Vassiliou and Andreas Pitsillides
Department of Computer Science, University of Cyprus

Prabagarane Nagaradjane
Wireless Technology Lab, Department of ECE, Sri Sivasubramaniya Nadar College of Engineering, Chennai-India

Iacovos Ioannou, Christophoros Christophorou and Vasos Vassiliou
CYENS Center of Excellence, Nicosia, Cyprus

Andreas Pitsillides
Department of Electrical & Electronic Engineering Science, University of Johannesburg

network operation is required. Furthermore, the UEs are clustered based on their Euclidean distance using Mean-Shift, Affinity Propagation, OPTICS, K-Means, Spectral Clustering, Agglomerative Clustering and BIRCH unsupervised ML approaches, and the devices are categorized based on their eligibility to become a UE-VBS for the corresponding cluster using Random Forest, AdaBoost, and Gradient Boosting supervised ML classification approaches. Then, using a heuristic algorithm, we determine the optimal cluster head (CH) by exploiting the findings of our classification model. The proposed framework is simulated for a nonuniform distribution of the UEs in time and space and quantified using statistical analysis. Our simulation results demonstrate that the proposed model achieves an accuracy of around 95% using Random Forest classifier and the K-Means clustering.

Keywords Classification · Clustering · Small Cells · Two-stage Machine Learning Engine · Virtual Base Station · 5G · 6G

1 Introduction

The deployment of 5G network is still in its early phases, and research is still on to accomplish the ambitious goals of lower energy consumption, higher data rates, seamless connectivity, high bandwidth, minimal delay, as well as the capacity to accommodate more users. Moreover, minimizing network congestion, traffic control, providing better coverage, and service requirements like high data throughput force the adoption of small cells (small base station) under the macro BS [6,7]. As a result, the deployment of a large number of small base stations (SBSs) and the construction of small cells, everywhere, that support all types of radio access technologies (RATs) is inevitable in achieving the above-said objectives [1,2]. However, small cell deployment at large scales such as in emerging ultra-dense HetNet environments brings major deployment obstacles [3] that must be overcome before they can be deployed in the emerging 5G networks so that their full benefits can be realized. To be specific, the obstacles can be mainly related to i) The huge Capital Expenditure (CAPEX) investments needed by the mobile network operators (MNOs) to deploy a large number of small base station equipment to realize small cells; ii) The need for a large number of sites acquisition needed by the MNOs for mounting their BS equipment as well as the huge time and economic costs associated with their deployment; iii) The changing mobile traffic dynamics[1] currently not efficiently addressed in the existing small cell deployments due to the static positioning constraints, and iv) The higher economic costs related to the Operation and Maintenance (O&M) of a large number of BS equipment.

To address the aforesaid challenges in ultra-dense 5G cellular networks [10], a virtual base station (VBS) is presented as an approach in [8], which allows for the easy and flexible deployment of virtual small cells in a mobile environment. With the VBS approach, smartphones of the general population are augmented with the base station and relay node functionalities and implanted as a versatile VBS that complements the infrastructure of MNOs. Based on network needs, user equipment (UE), i.e, the smartphone enhanced with BS functionalities, henceforth

[1] Related to the dynamic activity and mobility of the human factor as well as the masses location preferences that change through time, making mobile traffic dynamic and asymmetrically distributed in both spatial and time domains.

referred to as a UE-VBS can be selected to serve as i) a virtual small cell (VSC) used for coverage/capacity/data rate expansion in stressed hot spot areas where the infrastructure is weak; and/or ii) an intermediate virtual relay node (VRN) in a communication path facilitating the effective and efficient flow of data within the radio access part [11]. A UE-VBS acting as VSC or VRN can also share its link with other UE-VBSs or UEs in its vicinity allowing them to directly communicate without the requirement of a communication link through a macro base station (as shown in [21]).

Note, that there is an increasing interest, both from the research community as well as by the industry in developing practical small cell solutions that will address the above-mentioned challenges and drive the integration and commercialization of small cells into 5G that will be flexible and dynamic to handle different citations. Also, in the latest 3GPP specs [22], self-organizing networks (SON) solutions are introduced in order to reduce the operating expenditure (OPEX) and CAPEX that are mainly associated with the rapidly increasing numbers of small base stations in the network.

To this purpose, a new dynamic, highly adaptable, and cost-effective virtual UE-VBS layer can be established at the bottom layer of the Evolved UMTS terrestrial radio access network (E-UTRAN) architecture, leveraging the dense and ubiquitous distribution of mobile users (owners of eligible UEs) currently existing in our physical world. On this virtual UE-VBS layer, the MNOs would be able to control and exploit on-demand, at any time and at any of the locations, the UEs that can serve either as VSCs or VRNs. Furthermore, as the UEs are carried by the users, mobility of the users implies mobility of the UE-VBSs, and hence, the major challenges related to small cell deployment can be vastly eliminated. More precisely: i) CAPEX is offloaded to the mobile users whose UEs (i.e., smartphones) can be exploited as virtual BS equipment; ii) There is no need for sites acquisition or expensive installations since the UE-VBSs are "mounted" on the mobile users; iii) Any constraints related to mobile traffic dynamics are removed since the UEs that act as UE-VBSs are carried by the mobile users, that as part of the crowd will always follow the crowd's location trends and location preferences, thereby, extending capacity in a targeted and scalable manner, and thus, resulting in achieving higher revenues for the MNOs, and iv) The economic costs related to O&M can be significantly reduced since the mobile users, being the owners of the UE-VBS equipment will be responsible for their operation and maintenance.

This work builds on the principles of the UE-based virtual base station (UE-VBS) concept [8,9] and targets the major challenges that need to be addressed for maximizing their potentials in developing 5G ultra-dense networks and at the same time act in the same way that small cells do. More specifically, in this work, we aim to propose and develop a two-stage machine learning (ML) engine that can be used to activate UE-VBSs at the "correct"[2] locations at the time of need, for dynamic VSC or VRN deployments, thereby, extending data rates, capacity, and backhauling capabilities that result in enhanced service quality across the cell. Moreover, our proposed solution divides the process of selecting a UE-VBS into two stages: i) clustering UEs based on their Euclidean distance from the BS;

[2] That is close to where the consumption of data is taking place. Getting access to huge numbers of sites in the "correct" most profitable in terms of Sum Rate location, and in a cost-effective manner, is key to a commercially viable dense HetNet.

and ii) binary classification of UEs in each cluster into eligible and ineligible UEs based on the devices' quality of service (QoS), quality of experience (QoE), data rate, distance from the BS, battery, and processing power. Finally, a candidate UE from the list of eligible UEs is chosen randomly to be activated as a UE-VBS (cluster head). The proposed model aims to dynamically identify a UE-VBS node through which aggregated mobile data traffic can be sent in order to ensure QoS, maintain high data rates, and simultaneously take advantage of today's UEs' superior capabilities.

In this work, we use competitive well known unsupervised ML techniques like K-means, Spectral clustering, Agglomerative clustering, Birch, Mean shift, affinity propagation, and OPTICS (shown in Section 5) to find which of these ML techniques is optimum to implement clustering. From our analysis (shown at Section 5.2), we inferred that K-means algorithm results in the best performance with a silhouette coefficient[3] of 0.46, and hence, it is chosen to device clustering in the first stage. In the second stage, binary classification is performed with the use of supervised ML competitive techniques like AdaBoost, Gradient boosting, and Random forest. In the context of classification, the best performance is seen to be provided by a Random forest classifier with an F1 score of 0.86 and a Recall score of 0.80, demonstrating its applicability in determining the eligible UEs that will be activated as UE-VBSs. Finally, a heuristic algorithm is used in order to select one eligible device as UE-VB for each cluster.

The key contributions of this paper are summarised below:

- We propose a two-stage ML approach for the UE-VBs selection and network augmentation.
- We investigate the various clustering approaches in the first stage to choose the best approach to cluster the UEs that can serve as UE-VBSs.
- We analyze in the second stage the binary classification of the devices to become eligible UE-VBs and non-eligible devices using various supervised ML competitive techniques.
- We propose a heuristic algorithm to select a UE-VBS for each of the clusters based on the results of clustering and classification carried out in the first stage and second stage, respectively.
- We show that our proposed approach performs better than random selection in terms of the achievable sum-rate.

Following this introductory section, the rest of the paper is organized as follows: Section 2 provides some background information related to our investigation and also it analyses the existing research contributions that are related to our work. Section 3 presents our problem description. Section 4 details the simulation set up followed by how the UE configuration information is processed and merged to construct the training dataset, examines the performance indicators, and defines the features/parameters for selecting the eligible UEs. Also, we brief out the selection of UE-VBS from the eligible UEs. Section 5 describes the investigated

[3] The Silhouette coefficient is used to evaluate the quality of clusters created using clustering algorithms such as K-means in terms of how well samples are clustered with other samples that are similar to each. The value of the Silhouette score varies from -1 to 1. If the score is 1, the cluster is dense and well-separated from other clusters. A value near 0 represents overlapping clusters with samples very close to the decision boundary of the neighboring clusters. A negative score [-1, 0] indicates that the samples might have got assigned to the wrong clusters.

clustering approaches, as well as examines how the approaches are trained, tested, and evaluated using the created dataset. Section 6 explains the investigated classification approaches. Additionally, it examines how these classifiers are trained, tested, analyzed, and compared by the created dataset. Section 7 expounds the heuristic selection algorithm that we use to choose a UE that will become the cluster head in a specific cluster among a number of eligible UEs obtained from the classification approach. Finally, Section 8 draws the concluding remarks and our future directions.

2 Background And Related Work

In this section, we provide the background information and related work on 5G networks, especially in the context of UE-VBS selection and deployment, small cells selection, and ML-based approaches aiming to create ultra-dense networks. Following the related work, we conclude with a brief description and a comparison of the techniques used in our work, contrasted with some papers in the related work. In this subsection, we present the background information regarding UE-VBS selection, deployment, and ML. Additionally, we discuss the related work of clustering and classification approaches in telecommunication.

2.1 Background Work Related to UE-VBS

The small base stations (SBS) in the context of femtocells, picocells, and nano cells is usually deployed in a static environment based on the demand. In particular, SBSs are deployed in places around the hotspots with the aid of static information. Nevertheless, fixed small cells deployment can be inefficient in scenarios where there can be an unpredictable surge in traffic. Hence, in order [13] to cope with the mobile traffic dynamics in an ultra-dense 5G cellular network, a UE-VBS is proposed as a solution in [8], which assist in the effectual deployment of a massive number of VSC in real-time in an effortless and flexible manner. The aspects of the UEs the functioning as UE-VBSs is presented in [12], [9] and [11]. In [12], authors considered a simple scenario where the BS has limited capacity, servicing connections, bandwidth as well as power and is unable to fulfill the demands and requirements of a number of client UEs. In order to address these problems, the authors have proposed that a BS can select a subset of the UEs that are being served (predefined as capable and willing to offer their services) to serve a small neighborhood of UEs in their proximity by becoming UE-VBSs. Furthermore, this research contribution assumes that the subset of UEs chosen have the required hardware and software specification to become the eligible UE-VBSs. More specifically, the selection of a UE to become an eligible UE-VBS is based on the distribution of UEs and that, the number of UEs that will associate with the active UE-VBS based on the maximum received power.

In [9], an initializing matching connection algorithm (IMCA) has been proposed. The IMCA targets the implementation of the initial connection process between client UEs and eligible UEs (that will become VSCs) based on the SINR preference, resulting in the formation of clusters. Then, the IMCA algorithm op-

timizes the clusters based on the upper and lower bound constraints along with the size of the cluster that leads to the formation of a UE-VBS.

Affinity propagation clustering (APC) approach is implemented in [11], aiming the selection of the most appropriate UEs from a set of UE-VBSs to supplement an overlay loaded SBS. The paper combines two diverse fields of computers, mobile networks, and machine learning. By adopting an ML approach, this work dynamically determines a subset of UEs served by the macro BS that can be activated to serve as a UE-VBS in their vicinity. Also, the APC tackles the problem of classification of the devices so that a device can be chosen to act as a small cell UE-VBS, which is not addressed in open literature despite the fact that significant contributions relating to clustering of the UEs exist. The algorithm reported by the authors involves organizing objects into groups, whose members are similar by considering all data points as probable candidate cluster centers, and then it selects the UE-VBS for each of the clusters based on the euclidean distance and transmission range (Min and Max).

2.2 Background Work Related to AI/ML

With the advent of ML, devices or things are becoming automated, requiring less labour and hence less manual control. In some cases, ML is also proving to be more efficient and reliable than statistical methods, revolutionizing several research problems, thereby leaving very few uninfluenced in the digital world. Moreover, several techniques with many innovations and revamps have so far been employed to accomplish artificial intelligence (AI) and ML tasks. Nonetheless, two general categories of ML algorithms aid this research: unsupervised learning and supervised learning . In the following sections we will examine the: i) clustering techniques related to unsupervised learning; and ii) classification techniques related to supervised learning.

2.2.1 Background Work Related to ML Clustering with Unsupervised Learning

Unsupervised learning [31] is a machine learning (ML) technique that focuses on pattern recognition issues and uses a set of input vectors x without any matching target values as training data. Unsupervised learning is a technique for extracting references from datasets that contain input data but no labelled answers. It is a method for identifying significant structures, explaining underlying processes, generating traits and groups in a set of samples. The purpose of clustering is to find groups of comparable examples within the data or, in the case of density estimation, to establish how the data is distributed in space. Clustering is partitioning a population or set of data points into several groups so that data points in the same group are more similar to each other and dissimilar to data points in other groups. It is essentially a grouping of items based on their similarity and dissimilarity.

K-Means Clustering Algorithm The K-means clustering algorithm [38] is an unsupervised ML technique used to identify clusters of data objects in a dataset. Here, the data objects are UEs, and the data set is location information. There

are multiple clustering algorithms, but K-means is one of the oldest and most approachable. It is also important to cluster UEs into non-overlapping groups which can be achieved using K-means. K-means clustering algorithm uses distance (Euclidean) as the metric and number of clusters as the parameter. The steps involved in the K-means clustering algorithm are:

1. Specify number of clusters K.
2. Initialize centroids by first shuffling the dataset and then randomly selecting K data points for the centroids without replacement.
3. Keep iterating until we don't see a change in the centroids, i.e., assignment of data points to clusters isn't changing.
4. Compute the sum of the squared distance between data points and all centroids using Euclidean distance formula is given by

$$\hat{d} = \sum_{i,j=1}^{\bar{N}} \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} \tag{1}$$

   Equation describes the Euclidean distance formula where $\hat{d}$ is the Euclidean distance, $\bar{N}$ is the number of clusters and x_i, x_j, y_i, y_j are UE coordinates.
5. Assign each data point to the closest cluster (centroid).
6. Compute the centroids for the clusters by taking the average of all data points that belong to each cluster.

The optimum number of clusters $\tilde{k}$ is extremely essential as it is the only parameter considered for K-means clustering algorithm. This can be determined by using the Elbow method. To perform the elbow method, K-means is run several times with $\tilde{k}$ value incremented for each iteration, and the sum of the squared error (SSE) is recorded. The value at which the SSE dips is referred to as elbow and is considered to be the optimum value for the number of clusters.

Note that, in our approaches the distance between the different UEs is computed using the Euclidean distance formula which only requires the two-dimensional coordinates, i.e., the (x, y) coordinates of all the UEs in the data set. Additionally, the optimal number of clusters for a given data set is obtained either by using the elbow method (Silhouette score) or, mean shift clustering.

Mean-Shift Algorithm On the contrary to unsupervised learning, the mean-shift [39] is a clustering algorithm that iteratively assigns the data points to the clusters by altering the points towards the mode, where, in the context of mean-shift, the mode is the highest density of data points in the region. As a result, it is also referred to as the mode-seeking algorithm. To be specific, the algorithm iteratively assigns each of the data points towards the centroid of the closest cluster and the closest centroid's direction is determined by where the majority of the points are nearby at. Therefore, after each iteration, each of the data points will move towards where most of the data points are thereby leading to the cluster center. Once the algorithm ends, each of the points will be assigned to a cluster. Unlike the K-means, mean-shift does not need the knowledge of the number of clusters in advance and the number of clusters is determined pertaining to the data by the algorithm. However, the computational cost of the mean shift is high and is of the order of $O(n^2)$ [33]. In the mean shift clustering algorithm, the first step is

to represent the data as points. Mean-shift works on the concept of kernel density estimation (KDE). Suppose, if the data is sampled from a probability distribution, then KDE is an approach to estimate the distribution of the set of data, which is called the probability density function. The KDE works by assigning a kernel on each of the points in the data set. A kernel is a weighting function that is generally used in convolution. Many different types of kernels exist, but the most sought after is the Gaussian kernel. By adding the individual kernel yields a probability surface example density function, and depending on the kernel bandwidth employed, the resultant density function may vary.

The steps involved in mean-shift clustering algorithm are as follows.

1. Initialize random seed and window W.
2. Calculate the centre of gravity (mean) of W.
3. Shift the search window to the mean.
4. Repeat Step 2 until convergence. KD estimator for the full population's density function is given by,

$$f_k(u) = \frac{1}{\hat{n}\bar{h}^{\bar{d}}} \sum_{i=1}^{\hat{n}} \bar{K}\left(\frac{u - u_i}{\bar{h}}\right) \tag{2}$$

where, $\hat{n}$ denotes the identically distributed samples that are independent, u_i is the set of points in $\bar{d}$-dimensional space of the given dataset sampled from some larger population, $\bar{K}$ is the chosen Kernel function having bandwidth parameter $\bar{h}$. The kernel function, $\bar{K}$ here is required to satisfy the following two conditions:

1.
$$\int \bar{K}(u)du = 1 \tag{3}$$

2.
$$\bar{K}(u) = \bar{K}(|u|), \forall u \tag{4}$$

Two popular kernel functions that satisfy these conditions are given by:

1. Flat/Uniform
$$\bar{K}(u) = \frac{1}{2}\begin{cases} 1 & \text{if} -1 \leq |u| \leq 1 \\ 0 & \text{else} \end{cases} \tag{5}$$

2. Gaussian
$$\bar{K}(u) = \frac{1}{(2\pi)^{\frac{\bar{d}}{2}}} e^{-\frac{1}{2}|u|^2} \tag{6}$$

Spectral Clustering Spectral clustering [40] is a clustering algorithm that is widely used in multivariate statistics. It treats each data point as a graph node and thus transforms the clustering problem into a graph-partitioning problem. Firstly, we build the similarity graph in the form of an adjacency matrix which is represented by $\mathcal{A}$. The adjacency matrix can be built using the epsilon-neighborhood graph. The parameter epsilon is fixed beforehand, and each point is connected to all the points which lie in the epsilon radius.

The data points are then projected onto a lower dimension. This step is done to account for the possibility that members of the same cluster may be far away in the given dimensional space. This translation to a lower dimension is achieved

by computing the graph Laplacian matrix. To compute it though first, the degree of a node needs to be defined. The degree of the ith node is given by

$$\tilde{d}_i = \sum_{j=1|(i,j)\epsilon E}^{n} w_{ij} \tag{7}$$

where w_{ij} is the edge between the nodes i and j as defined in the adjacency matrix above. The degree matrix is defined as follows:

$$\tilde{\mathcal{D}}_{\mathbf{ij}} = \begin{cases} \tilde{d}_i, i = j \\ 0, i \neq j \end{cases} \tag{8}$$

Thus the graph Laplacian matrix is defined as:-

$$\mathcal{L} = \mathcal{D} - \mathcal{A} \tag{9}$$

This matrix is then normalized for mathematical efficiency. To reduce the dimensions, first, the eigenvalues and the respective eigenvectors are calculated. If the number of clusters is $\bar{N}$ then, the first eigenvalues and their eigenvectors are taken and stacked into a matrix such that the eigenvectors are the columns. Finally, we cluster the reduced data by using any traditional clustering technique–typically, K-means clustering in this work.

Agglomerative Clustering Agglomerative clustering is also known as hierarchical agglomerative clustering (HAC) [44]. This algorithm is a bottom-up algorithm. This algorithm builds a structure that is more informative than the unstructured set of clusters returned by traditional clustering algorithms. This clustering algorithm does not require to prespecify the number of clusters. Being a bottom-up algorithm, it treats each data as a singleton cluster at the outset and then successively agglomerates pairs of clusters until all clusters are merged into a single cluster that contains all data. This algorithm uses the linkage distance parameter, i.e., a metric that is used to compute the linkage between data points. This metric can be "euclidean", "manhattan", "cosine", or any pair-wise distance metric. The algorithm starts with each data point as a cluster and recursively merges the clusters based on the linkage distance. The linkage criterion determines the distance between sets of observations as a function of the pairwise distances between observations. The following are some of the linkage criterion used:

$$\max \{\, d(a, b) : a \in \mathbb{A},\, b \in \mathbb{B} \,\}. \tag{10}$$

for a global linkage,

$$\min \{\, d(a, b) : a \in \mathbb{A},\, b \in \mathbb{B} \,\} \tag{11}$$

for a single linkage,
where, $\mathbb{A}$ and $\mathbb{B}$ are the two sets of observations, and the linkage criterion determines the pairwise distances between these sets of observations.

Affinity Propagation The Affinity propagation [43] technique creates clusters without the knowledge of the number of clusters. Unlike clustering algorithms such as K-means or K-medoids, the Affinity propagation calculates clusters with the utilization of messages passing, i.e., by sending messages between data points until convergence. This technique requires the following two inputs: i) preference, which controls how many exemplars (or prototypes) are used; ii) damping factor, which damps the responsibility and availability of messages to avoid numerical instabilities when updating these messages.

The Affinity Propagation dataset is defined using a small number of exemplars, and 'exemplars' are participants of the input set representing clusters, which are used for clustering. More precisely, the messages sent between pairs correspond to the suitability for one sample to be the exemplar of the other, that is updated in response to the values from other pairs. Thus, the updating happens iteratively until the convergence point is attained. At the convergence point, the final exemplars are chosen, and hence the final clustering is obtained.

The responsibility matrix `R` has values $r(i,k)$ that compute how well-suited x_k is to serve as the exemplar for x_i, relative to other candidate exemplars for x_i. The availability matrix `A` contains values $a(i,k)$ that represent how "suitable" it would be for x_i to pick x_k as its exemplar by taking into account other points' preference for x_k as an exemplar. Both matrices are initialized to all zeroes and can be viewed as log-probability tables. The algorithm then performs the following updates iteratively

To start with, the responsibility updates are made available for all the data points as shown below.

$$r(i,k) \leftarrow s(i,k) - \max_{k' \neq k} \left\{ a(i,k') + s(i,k') \right\} \tag{12}$$

Then, the availability matrix is updated

$$a(i,k) \leftarrow \min \left(0, r(k,k) + \sum_{i' \notin \{i,k\}} \max(0, r(i',k)) \right), \quad \text{for } i \neq k \tag{13}$$

and

$$a(k,k) \leftarrow \sum_{i' \neq k} \max(0, r(i',k)) \tag{14}$$

Note that, iterations are performed until either the cluster boundaries remain unchanged over a specific number of iterations or if a specific predetermined threshold number (of iterations) is reached.

BIRCH Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) [41] is a clustering algorithm that can cluster large datasets by first generating a small and compact summary of the large dataset that retains as much information as possible. This smaller summary is then clustered instead of clustering the larger dataset. Clustering algorithm like K-means requires the number of clusters as input, and if this parameter is not given it does not perform the clustering very efficiently, and further, it is difficult to process large datasets with a

limited amount of resources (like memory or a slower CPU). So, regular clustering algorithms do not scale well in terms of run time and quality as the size of the dataset increases. The BIRCH algorithm has a runtime complexity of $O(n)$. BIRCH summarizes large datasets into smaller, dense regions called clustering feature (CF) entries. We compute the CF for our datapoints, given a set of $\mathbb{N}$ d-dimensional data points, CF of the set is defined as the triplet $CF = (\mathbb{N}, \overrightarrow{\mathbb{LS}}, \mathbb{SS})$, where $\overrightarrow{\mathbb{LS}} = \sum_{i=1}^{\mathbb{N}} \overrightarrow{\mathbb{X}_i}$ is the linear sum and $\mathbb{SS} = \sum_{i=1}^{\mathbb{N}} (\overrightarrow{\mathbb{X}_i})^2$ is the square sum of data points. Once we compute the CF values, we construct the CF tree. The CF tree is the actual compact representation of the cluster features. A CF tree is a tree where each leaf node contains a sub-cluster. Every entry in a CF tree contains a pointer to a child node and a CF entry is made up of the sum of CF entries in the child nodes. There is a maximum number of entries in each leaf node. This maximum number is determined by the hyper-parameter threshold.

OPTICS OPTICS [42] or ordering points to identify the clustering structure is a density-based clustering algorithm. OPTICS requires two parameters: ε, which describes the maximum distance (radius) to consider, and MinPts, describing the number of points required to form a cluster. A point p is a core point if at least MinPts points are found within its ε-neighborhood $\mathcal{N}_\varepsilon(p)$ (including point p itself). ε is the parameter specifying the radius of a neighborhood with respect to a point. The value for ε can then be chosen by using a k-distance graph, plotting the distance to the $k = minPts - 1$ nearest neighbor ordered from the largest to the smallest value. OPTICS computes two parameters for clustering, namely core-distance and reachability distance.

- **Core Distance (CD):** It is the minimum value of radius required to classify a given point as a core point. If the given point is not a core point, then it's core distance is undefined.

$$\mathrm{CD}_{\varepsilon, MinPts}(p) = \begin{cases} \text{null} & \text{if } |\mathcal{N}_\varepsilon(p)| < MinPts \\ \min(\mathcal{N}_\varepsilon(p)) & \text{otherwise} \end{cases} \tag{15}$$

- **Reachability Distance (RD):** It is defined with respect to another data point q. The RD between a point p and q is the maximum of the core distance of p and the euclidean distance(or some other distance metric) between p and q. Note that, the RD is not defined if q is not a core point, and RD of an alternative point, say o from a point p can be experessed as

$$\mathrm{RD}_{\varepsilon, MinPts}(o, p) = \max(\mathrm{CD}_{\varepsilon, MinPts}(p), \mathrm{dist}(p, o)) \tag{16}$$

2.2.2 Background Work Related to ML Classification with Supervised Learning

Supervised learning [23] is the category of ML algorithms that works on pre-labelled data. This means there are pre-defined inputs, which we call features, and pre-defined output(s) called label(s). Cases where data are continuous and do not belong to a few classes fall under regression. When the output label is/are discrete class (es), it is classification. Unsupervised learning algorithms [24] use

unlabeled data. These models work in a way that they first learn the features from the fed data, and then the validation step is carried out, following which, data are tested for the already learned features. The Classification algorithm is a Supervised Learning technique that uses training data to determine the category of new observations. Classification is the process of software learning from a dataset or observations and then classifying fresh observations into one of several classes or groupings (e.g., Yes / No, 0 / 1). Targets/labels or categories are all terms used to describe classes.

Random Forest Random Forest [36] is a set of the decision tree. Each tree in the ensemble is built using the random forest algorithm from a sample drawn with replacement (i.e., a bootstrap sample) from the training set. It creates an uncorrelated forest of decision trees by combining bagging and feature randomness. Each decision tree is a set of internal nodes and leaves. In the internal node, the selected feature is used to make a decision on how to divide the data set into two separate sets with similar responses within. Furthermore, when splitting each node during tree construction, the best split is determined by selecting either all input features or a random subset of size. The features for internal nodes are selected with some criterion, which for classification tasks can be Gini impurity or information gain. We can measure how each feature decreases the impurity of the split (the feature with the highest decrease is selected for the internal node). For each feature, we can collect how on average it decreases the impurity. The average number of overall trees in the forest is the measure of feature importance. The feature importance calculated by Random Forest for our model is illustrated in Fig. 5. The goal of these two randomness sources is to reduce the variance of the forest estimator. Individual decision trees, in fact, have high variance and tend to overfit the model. Forests with injected randomness produce decision trees with somewhat decoupled prediction errors. Some errors can be eliminated by taking an average of those predictions. Random forests reduce variance by combining diverse trees, sometimes at the expense of a slight bias increase. In practice, the variance reduction is frequently significant, resulting in a better overall model.

Adaptive Boosting (AdaBoost) The adaptive boosting algorithm or AdaBoost [35], is based on the ensemble method of boosting, in which the weights are re-assigned to each instance with higher weights assigned to instances that are incorrectly classified. AdaBoost's core principle is to fit a series of weak learners (models that are only slightly better than random guesses, such as small decision trees) on repeatedly modified versions of the data. To produce the final prediction, the predictions from all of them are combined using a weighted majority vote (or sum). At each so-called boosting iteration, the data is modified by applying weights to each of the training samples. Initially, the algorithm simply trains a weak learner on the original data. The sample weights are individually modified for each iteration, and the learning algorithm is reapplied to the re-weighted data. At each step, the weights of those training examples that are incorrectly predicted by the boosted model induced in the previous step are increased, while the weights of those that are correctly predicted are decreased. As the iterations progress, the weights that are difficult to predict, gain more and more clout. As a result, each subsequent weak learner is forced to focus on the examples that the previous ones in the sequence missed.

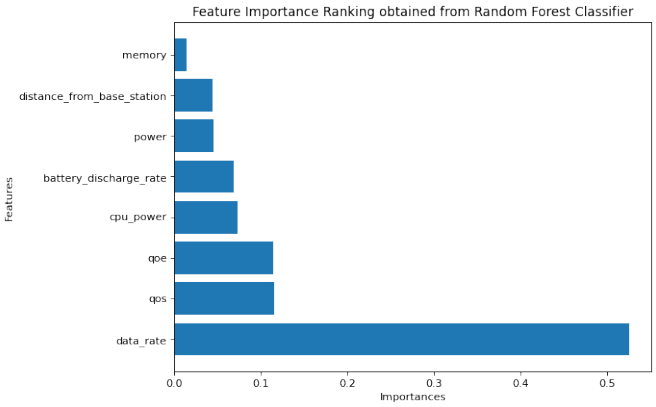


Fig. 1: Feature importance calculated by Random Forest based on the Gini importance.

Gradient Boosting Gradient boosting [37] is a type of boosting used in machine learning. It is based on the assumption that when the best possible next model is combined with previous models, the overall prediction error is minimized. To minimize error, the key idea here is to set the target outcomes for the next model. The target outcome for each case in the data is determined by how much change in the prediction for that particular case affects the overall prediction error. If a small change in a case's prediction results in a large reduction in error, the case's next target outcome is set to a high value. The error will be reduced by predictions from the new model that is close to their targets. If a small change in a case's prediction results in no change in error, the case's next target outcome is set to zero as the changes to this prediction have no effect on the error. Gradient boosting gets its name from the fact that each case's target outcomes are determined by the gradient of the error with respect to the prediction. In the space of possible predictions for each training case, each new model takes a step in the direction that minimizes prediction error.

2.3 Related Work

In our work, we utilize clustering (unsupervised) and classification (supervised) to develop an ML model for the selection of UE-VBSs, thereby, forming the network dynamically using VSCs. Therefore, in this section, we present the related work that is in line and most relevant to the research problem investigated in this paper.

In [25], the authors examine the technology and opportunities for incorporating AI into the design of autonomous wireless systems. Unlike data-driven approaches, where knowledge-discovery supported by ML techniques is used for prediction in matching problems, the authors' here, have provided the readers with motivation and general AI methodology of autonomous agents in the context of self-organization in real-time by unifying knowledge management with sensing, reasoning, and active learning. Additionally, the authors have extended the concept of machine intelligence to wireless systems by summarising the properties

of training-free and training-based methods in wireless environments, with reinforcement learning as a major representative of AI. The principles of knowledge management are highlighted followed by a functional example of an autonomous agent.

In [26], authors proposed two power control (PC) algorithms based on ML, distributed Q-learning, and decision tree classifier (CART) for D2D communication in underlay cellular networks. These algorithms have been compared with the traditional PC methods in terms of computational complexity, communication throughput, and energy efficiency to provide insight into the potential of combining ML and wireless communications. Also, the approach targets the optimization problem that maximizes the sum of all users' logarithmic utility functions, and it is divided into two major sub-problems: Firstly, the PC for a single D2D pair and potential reuse candidates in order to maximize the overall throughput with the goal of maximizing either system capacity or energy efficiency while at the same time protecting cellular users' QoS is addressed. Secondly, proportional fair scheduling of resources for multiple D2D Pairs and cellular users is discussed that improves system fairness with a data-driven decision tree classifier algorithm trained with the results obtained from the first sub-problem. The performance of the proposed algorithms evaluated in terms of both system capacity and energy efficiency have been shown to provide better results in comparison to traditional closed-loop and open-loop power control algorithms.

The innovating work reported in [27] proposes the use of a distributed AI framework with the use of Belief-Desire-Intention eXtended (BDIx) agents and a transmission mode selection approach/plan called "DAIS" for a D2D communication network that maximizes the data rate and minimizes the power consumption in the network while taking into account the computational load. The framework presented by the authors has been realized with the extension of the BDI agents to BDIx agents with ML and reinforcement learning, and the simulations results demonstrate the utility of BDIx agents in solving issues related to D2D problems. Furthermore, this article reports that link establishment is an optimization problem that should be solved in a distributed fashion using AI rather than as a global problem that must be solved centrally so that communication links can be established in less time. Besides, the authors show that distributed AI is the one of the bests solutions for optimal link realization in D2D communication.

The contributions in [28] are aiming to explore strategies for improving spectral efficiency by invoking small cells, coordinated multipoint, and massive MIMO. In this study, the authors have adopted the strategy of successive selection of UEs, wherein, one UE is selected and all of the nearby UEs are grouped together. Suppose, if the UE becomes ineligible to act as a VBS (this is found using a feedback mechanism) the nearby UE is chosen, and the process continues until an eligible UE is found. This article also justifies the benefits of integrating small cells with coordinated multipoint and massive multiple input multiple output (MIMO) with feasibility and synchronization tests. From the results obtained it has been demonstrated that the spectral efficiency has improved by 100 percent based on their proposed scheme. Finally, it is concluded that with reasonable effort, an increase in mobile traffic up to 1000 times is possible using a smart combination of all the techniques presented in this work.

In [29], authors analyzed the performance of small cells using unmanned aerial vehicles (UAVs), and also have extensively listed the advantages and challenges

such as efficient deployment, power consumption, coverage optimization, and interference management while using drones for the deployment of UEs. In order to address these challenges, the authors have extensively delved into the various ground-to-air channel models with multiple cases of interference and derived expressions for the same. Their results have shown the existence of an optimal drone small cells (DSCs) separation distance that results in maximum coverage for a given target area and also, provides a stepping stone to address the more general cases that constitute a large number of DSCs.

A multi-variant clustering approach with interference alignment to improve the spectral efficiency of densely packed small cells is introduced in [30]. An experiment has been conducted for a small cell network that comprises a maximum of 50 small cells distributed in a 0.25 km^2 region with a transmit power of 10 dBm, path loss exponent of 3.76, and a maximum bandwidth of 180 kHz between transmitter and receiver pair. Optimal users are identified and classified as clusters that satisfy the maximum sum rate achievement with the consideration of user density and path loss. Based on this study, it is found that maximum bits are transferred because the proposed multi-variant clustering approach with interference alignment results in the suppression of path loss and interference which in turn contributes to the improvement in the spectral utilization.

In most of the previous works presented above, the selection of UEs was carried out using non-deterministic approaches. Against this background and based on the research gap identified in the open literature, in this paper, we propose a 2-stage machine learning pipeline to facilitate the selection of the UEs. In the first stage, we employ optimized clustering, followed by classification in the second stage. Our investigation reveals that the proposed classification model achieves 95% accuracy in classifying a UE to be an eligible candidate for serving as a UE-VBS or not. Also, the UE-VBS selected based on our proposal exhibit better network metrics (i.e., data rate, QoS, QoE). To be specific, a data rate improvement close to 500% is achieved (shown in Table 9) which is significantly higher than the achievable data rate reported in [28]. Additionally, in this work, we perform an in-depth analysis of the various clustering algorithms in order to utilize better clustering. Also, we expound on the rationale behind the ML model exploited in our work in the forthcoming discourse and quantify the same based on the results obtained.

3 Problem Description

The rapid development of new advanced smartphones, tablets, wearable devices, etc., that support voice, data roaming, and video streaming require fast access to the network with high quality of connectivity. Furthermore, these advancements create the need for more bandwidth than already provided by the traditional high-tower mounted base stations of the existing mobile network. Therefore, to tackle the aforementioned needs, the 5G cellular network, presents very demanding requirements such as introducing changes to all mobile network layers and also, the use of small cells for ultra-densification achievement. The ultra-densification [7] in 5G systems encompasses the concept of a heterogeneous network (HetNet) popularised in the 4G LTE system, which can build more layers of cells to boost capacity in order to handle the ever-growing user terminals and offload the congestion in the data flow at the 4G base stations. Picocells, femtocells, and dispersed antenna

elements are among the potential solutions for HetNets that will play an important role in improving system performance [12]. The use of small cells in heterogeneous cellular networks can also result in a significant increase in energy efficiency. Small cell base stations are currently being deployed to increase network capacity while keeping infrastructure and deployment costs low. As a result, by strategically placing microcells and relays within a range of the macrocell, they can greatly unload the macrocell and save energy while offering better coverage. These changes in the network architecture are particularly useful in situations where exceptionally high capacity and data rates are required, such as offices, retail malls, subway stations, and other public locations with a high user density.

The main objective of this paper is to develop a two-stage ML engine that can be used to activate UE-VBSs at places adjacent to where data consumption occurs dynamically, resulting in enhanced service quality across the cluster. The proposed solution divides the process of selecting a UE-VBS into 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 based on the device QoS, QoE, data rate, distance from the BS, battery power, and processing power. Finally, a UE is activated to become the UE-VBS (cluster head) based on a heuristic algorithm that evaluates the UEs from the classification results based on the data rate and signal quality (i.e., channel quality indicator (CQI)) from the surrounding devices. The model aims to dynamically identify a UE-VBS node through which aggregated mobile data traffic will pass in order to ensure QoS, maintain high data rates, and take advantage of today's UEs' superior capabilities.

Accordingly, in this paper, we consider the implementation of a mobile network that is constructed using VSCs. To be specific, our approach aims to group the UEs into clusters that conceive a network composed of VSCs. Then, it aims to select the eligible UEs qualified to become UE-VBSs and possible cluster heads of each cluster. The final goal of the approach is to select one of the eligible UEs as a UE-VB and a cluster head for each of the VSCs. This two-stage approach aspires to achieve higher data rates for enhanced mobile broadband (eMBB) for massive machine-type communications (mMTC) in a network that encompasses a large number of devices than the one provided by the operator and thus, satisfy some of the 5G requirements [32].

4 Simulation Setup And Features Selection

In this section, we provide the description of the experimental setup that involves the simulation of the UE-VBSs in a 5G network and the generation of the dataset for the classifier. Moreover, this section provides an in-depth explanation of the various parameters and their corresponding analytical expressions used as features for the classification of eligible and non-eligible UEs.

4.1 Simulation Setup and Scenario Description

We have conducted the experiment with a computing machine that has the following configurations; Intel Core i5-8265U, 256GB M.2 PCIe NVMe SSD, 8GB

DDR4 RAM, 2400 MHz, NVIDIA 4 GeForce MX150 (2GB GDDR5). The experimental setup is simulated using a python environment and implemented with the purpose to resemble a real-world scenario for extracting the relevant parameters that are required for the classification task. The UEs are spread in a non-uniform distribution around the base station. Multiple Snapshots of the UEs are taken with different population densities (of the UEs) to bring in a more comprehensive picture of the real-world scenario. The parameters of UEs like memory, instantaneous_battery_percentage, cpu_power were assigned randomly, and parameters like battery_discharge_rate, distance_from_base_station, recieved_power, signal interference noise ratio (SINR) are computed using the expressions presented in the subsequent section for all the data points of UEs.

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, conceive 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 with the use of competitive well known unsupervised ML techniques like K-means (as shown in [38]), Spectral clustering (as shown in [40]) , Agglomerative clustering (as shown in [44]), Birch (as shown in [41]), Mean shift (as shown in [39]), Affinity propagation (as shown in [43]) and OPTICS (as shown in OPTICS [42]), and in the second stage, binary classification is performed with the use of supervised competitive ML techniques like AdaBoost (as shown in [35]), Gradient boosting (as shown in Gradient boosting [37]), and Random forest (as shown in [36]). 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 K-means algorithm with a silhouette coefficient of 0.46, 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, in order to improve the model accuracy, the dataset used for training the Random Forest classifier is further improved by using the SMOTE approach in the unbalanced dataset, targeting to overcome the inherent bias exhibited as a result of the majority[4] class (eligible UEs). Finally, the model is subjected to statistical testing and analysis using SHAP and LIME. More details are provided in Section 6.5.

4.2 Dataset Generation and Features Selection For Clustering Techniques

The positions of UEs are initialized as cartesian pairs, i.e., (x, y) coordinates in space. The angle, speed, and time intervals are initialized randomly to facilitate the movement of UEs so that a dynamic selection environment can be realized. The UEs are mobile nodes and thus at various intervals, their positions and movements tend to be dynamic. The speed is randomly varied between 0 m/s to 10 m/s and the angle between 10 to 350 degrees and these changes occur at a time interval of the 20s (e.g., 20 s, 40 s, 60 s, ..). The simulation is run for multiple population densities as shown in Fig. 2, and the required parameters to be used are recorded.

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

Next, at a random time, a static snapshot of the coordinates is recorded and sent to the clustering algorithm to perform the clustering of the UEs at a specific time.

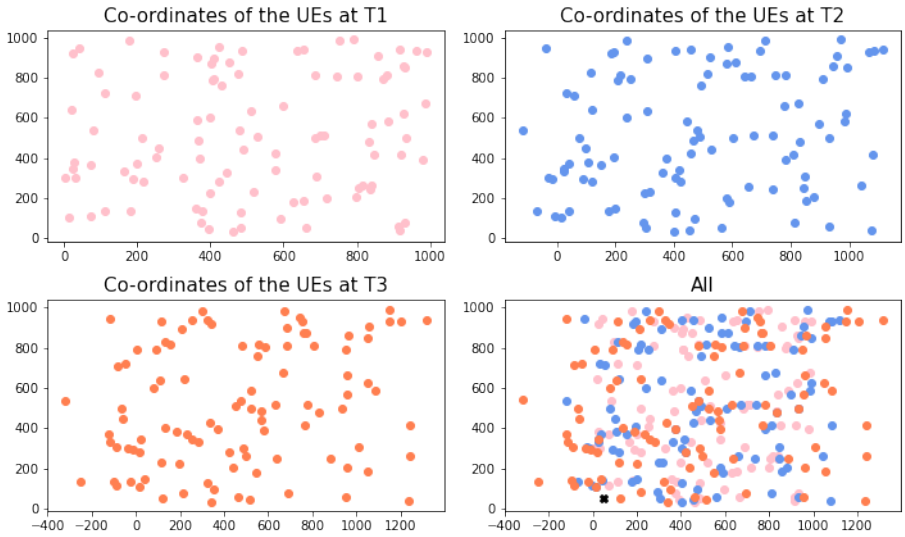


Fig. 2: Simulation of the movement of UEs in the spatial and temporal domain.

4.2.1 First Stage Clustering Data Used

For the preparation of the dataset for clustering, we used a random distribution function to generate the x and y co-ordinates of UEs at a particular time snapshot. The clustering was performed based on Euclidean distances.

4.3 Dataset Generation and Features Selection For Classification Techniques

The clustered UEs are classified as either non-eligible-UEs or eligible-UEs to become UE-VBSs. In the following paragraphs, we provide the network and UE parameters/features that are generated and computed using Python individually for all the UEs, which in turn will be used in the learning/training, testing, and evaluation steps of the utilized ML approache when the classification task is performed. More specifically, we make a brute force generation of all the possibilities in terms of the parameters shown below and then we evaluate as network engineers according to the features/parameters results if they are eligible UE-VBs or not. For the preparation of the dataset to aid training, testing (i.e., 70/30), and evaluation steps of the classifier, we need to compute the following parameters/features: i) dist_from_BS; ii) battery_discharge_rate; iii) power; iv) memory; v) cpu_power; vi) QoS; vii) QoE; and viii) data_rate. Additionally, this section provides the analytical expression related to the parameters considered. After clustering, the

process of inspecting the eligibility of UEs has to be carried out, i.e., the selection of UEs to act as UE-VBSs for each cluster. For selecting the eligible UEs, the parameters considered are:

1. Battery discharge rate
2. Received power
3. SINR
4. UE's memory, and
5. Processing Power of UE

Once all the above-mentioned parameters are decided, they are computed using the expressions given in the forthcoming discourse. These parameters are then used as network performance indicators for each UE in the clusters (as if it is to be chosen as an eligible UE).

1. Data Rate
2. QoS
3. QoE

Finally, the eligible UEs that will become UE-VBSs are selected following which, data cleaning[5] is performed.

4.3.1 Features Selection on the Second Stage that focusing on classification to eligible UEs and Not

In this section the selection of specific performance indicators that will result into features for the classification stage of selecting eligible UEs to become UE-VBS is executed.

Battery Discharge Rate Battery Power is one of the major factors in a wireless Ad hoc network. A node can transmit data to a longer distance only if it has sufficient battery power. Since most UEs are portable devices, they cannot often withstand the transmission of 5G signals for a longer period of time. When a device's battery dies, the network functionality gets disrupted, and hence, the battery discharge rate of the eligible UEs is an important factor in the selection of a suitable VBS, which is given by

$$Battery\ Discharge\ Rate = \frac{Capacity\ (mAh)}{Discharge\ Time\ (h)} \tag{17}$$

As can be seen from the above expression, the battery discharge rate is calculated as the ratio of instantaneous battery life, which is a value between 4000 to 7000 (in mAh), to the discharge time (in hours), which is a value in the range of 1 to 15 hours, and these values are assigned and calculated randomly. In a real-time scenario, these values can be obtained from the battery design information of each device.

[5] Data cleaning is the process of fixing or removing incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within a dataset.

Received Power The power received [11] by each UE in a cluster from the BS is a very important communication parameter, as this decides on how much power will be relayed to the other nodes in the cluster once it is assigned as a UE-VBS. When a signal travels over a channel, the signal power tends to fade due to the effects of the propagation channel and the power dissipates with distance, which is quantified in terms of the path loss $P_L(d)$ in dB given by

$$[P_L(d)]dB = [P_L(d_0)]dB + 10\alpha log_{10}\frac{d}{d_0} + \chi \text{ for } d_f < d_0 < d \tag{18}$$

where, α is the path loss exponent, χ is a Gaussian random variable, and d is the distance between T_x and R_x. $P_L(d_0)$ is the path loss at a reference distance d_0, which is calculated using Frii's free space formula given by

$$P_L(d_0) = P_r(d) = \frac{P_t G_t G_r \lambda^2}{4\pi^2 d^2 L} \tag{19}$$

where,

- P_t - Transmit Power
- $P_r(d)$ - Received power at a distance $'d'$
- G_t - Transmit antenna gain
- G_r - Receive antenna gain
- λ - Wavelength
- $L >= 1$ - System loss factor not related to propagation
- d- distance between the T_x and R_x.

The following values are considered for the received power (P_r) calculation in our experiment.

- $G_t = 1$
- $G_r = 1$
- $P_t = 43$ dBm (generally in the range 43-46 dBm)
- Pathloss distance exponent, $\alpha = 4$ (Obstruction due to building that includes shadowing)
- Frequency (FR1) = 5GHz
- $\lambda = 0.06$ m

Path loss represents signal attenuation as it propagates through space. Path loss is measured in dB and is the difference between the transmitted and received power. Note that, the received power predicted by path loss models is influenced by: i) reflection; ii) diffraction; and iii) scattering.

Signal to Interference plus Noise Ratio SINR has multiple important applications which include optimizing transmit power level for a target QoS, assisting with handoff decisions, and dynamically adapting the data rate for wireless applications [16]. The signal-to-interference-plus-noise ratio (SINR) of a link from UE_n to UE_m and a set of UEs $\{UE\}$ can be expressed as

$$\gamma_{n,m} = \frac{h_{n,m} P_n \lambda r_{n,m}^{-\alpha}}{\sigma^2 + \sum_{t \in \phi_m}^{\{t\}:t \neq n} h_{t,m} P_t r_{t,m}^{-\alpha}} \tag{20}$$

$n = 1, 2, ...N; m = 1, 2, ...M,$ and $t = 1, 2, .., T.$

where,

- $\{n\} \equiv \{m\} \equiv \{UE\}$ *and* $\{t\} \in \{UE\}$ *and* $t \neq n$ *and* $n \neq m$
- Channel gain of the desired user $h_{n,m}$
- Channel gain of the creating to UE_m interference device UE_t $h_{t,m}$
- $r_{n,m}$ is the distance of the link from UE_n to UE_m
- P_n is the transmit power of UE_n (depends on distance of UE, etc.)
- P_t is the transmit power of UE_t, which is the interfering UE
- $\{t\}$ denotes the set of devices that create interference to UE_m
- σ^2 is the noise power that is Additive White Gaussian Noise distribute (AWGN) and is assumed to be negligible as compared to the interference power and that the traffic follows a full buffer model.
- γ is the SINR

Data Rate/Sum Rate At UE, the data rate of each of the UEs is calculated using Shannon Capacity given by

$$R = Blog_{10}(1 + SINR)\, bps \tag{21}$$

where, R is the data rate, B is the channel bandwidth (100 MHz for 5G and 20 MHz for LTE) and SINR is the signal to interference plus noise ratio. In our work, we determine the sum-rate with the aid of the following expression.

$$TP = \sum_{i=1}^{c-size} B(1 + SINR_i) \tag{22}$$

where, c-size is the size of the cluster and $SINR_i$ is the SINR of the i^th cluster.

More specifically, we determine the achievable sum-rate for the proposed UE-VBS selection algorithm in the context of new radio (NR) frequencies defined for 5G.

Quality of Service To quantitatively measure QoS [18] a set of network parameters (i.e., packet loss, bit rate, throughput, transmission delay, availability, jitter, etc.) are often considered. QoS guarantees the network's ability to dependably run high-priority applications under limited network capacity. Using QoS, organizations will have the ability to optimize the performance of multiple applications on their network and gain visibility into the bit rate, delay, jitter, and packet rate.

QoS of the network is measured using the Jain's fairness index (JFI), which is expressed as

$$J(x_1,\, x_2, ...,\, x_{\bar{N}}) = \frac{(\sum_{i=0}^{m} x_i^2)}{\bar{N}(\sum_{i=0}^{m} x_i^2)} = \frac{\bar{\mathbf{x}}^2}{\overline{\mathbf{x}^2}} = \frac{1}{1 + C_v^2} \tag{23}$$

where $\bar{N}$ is the number of users in the system at a particular instance of time, C_v is the coefficient of variation of the sample, and x_i represents the throughput corresponding to the ith connection.

The spectral efficiency denoted as UE_iVBS_k of the link between the ith UE and its corresponding UE-VBS in the kth cluster is calculated by considering the link with the same FB between the jth UE and its corresponding UE-VBS denoted as UE_jVBS_n in the nth cluster. Thus, by considering the Eqs. 20 and 21, and by assuming the noise to be negligible, the spectral efficiency is given by

$$UE_iVBS_k = \sum_{n=1;n \neq k}^{N} log_2 \left(1 + \frac{p_i^k h_i^k}{p_{VBS,j}^n h_{VBS,j}^n}\right), \tag{24}$$

where the N is the number of clusters, p_i^k is the power received at the ith UE in the kth cluster, h_i^k is the channel gain of the link connecting ith UE and its CH ($UEVBS_k$). Also, the $p_{VBS,j}^n$ is the interference power received from the jth UE in the nth cluster, and $h_{VBS,j}^n$ is the channel gain of the link that connects the jth UE to its CH ($UEVBS_n$).

Quality of Experience The QoE fairness is used to assess the fairness among users by taking into account the QoE as realized by the end-user. This is an important metric that can aid in the network management where the service providers want to make their users amply satisfied (i.e. highest possible QoE) in a fair manner. In the literature, various approaches have been proposed to guarantee QoE fairness throughout the network, especially for applications like adaptive video streaming. In contrast to throughput, QoE is not quantified on ratio scales, and hence, Jain's fairness index cannot be used. Moreover, the measurement scale needs to be a ratio scale with a well-defined zero point. As a result, QoE may be measured on interval scales, say for example, on a 5-point mean opinion score, with 1 designating the lowest quality while 5 specifying the highest quality. Also, the coefficient of variation in the context of QoE is trivial, and therefore, the standard deviation σ may be utilized to provide a dispersion measure of QoE among users. In the [19] authors have introduced a QoE fairness index, which is given by

$$F = 1 - \frac{2\sigma}{H - L} \tag{25}$$

that considers the rating scale's lower bound L, H higher bound (i.e., Data Rate) and the standard deviation σ (provides a measure of the dispersion) of measurement metric.

The fairness ranges from $\frac{1}{N}$ to 1, with $\frac{1}{N}$ being the worst and 1 being the best. Absolute fairness (i.e., all users receive the equal allocation of the shared resources) is achieved when JFI = 1. The fairness is useful in identifying the underutilized channels and it is not excessively responsive to network flow patterns.

5 First Stage Approach for Selection: Analysis And Comparison of Clustering Approaches Investigated

Clustering is the task of separating the population or data objects into a number of groups (clusters) such that data points in the same groups are more similar. It is a collection of objects on the basis of similarity and dissimilarity between them. The first stage of the proposed machine learning pipeline is to cluster the UEs based on the distance from the base station. Clustering algorithms can be broadly classified by their ability to cluster the objects with/without having to know the number of clusters beforehand. Several clustering algorithms are tested out in our work to find the ideal algorithm for clustering the UEs. Specifically, the clustering algorithms considered are i) Mean-Shift, Affinity Propagation, and Ordering Points To Identify the Clustering Structure (OPTICS), which can achieve

clustering without having to know the number of clusters before (Table 1); and ii) K-Means, Spectral Clustering, Agglomerative Clustering, and Balanced Iterative Reducing and Clustering hierarchies (BIRCH) that have to know the number of clusters beforehand in order to achieve clustering (compared in Table 2).

In the subsequent sections, we describe the various clustering algorithms used in our analysis. Also, using silhouette scores, we identify the number of clusters that need to be formed for those approaches that need this input beforehand.

Overall, from the results provided in section 5.2, we can notice that compared to all other approaches K-means, followed by mean-shift provides the best results in terms of the silhouette score (0.46). However, the tradeoff here is that K-Means needs an extra step in the clustering process (i.e., needs to identify beforehand the number of clusters that needs to be formed), adding an additional overhead (i.e., CPU, Memory, time) in the computation.

Table 1: Comparison between different clustering algorithms that estimate the number of clusters

Clustering Algorithm	**Mean-Shift**	**Affinity Propagation**	**OPTICS**
Parameters	Kernal Bandwidth	Damping factor, Sample Preference	Minimum Cluster Membership
Metric	Euclidean Distance	Euclidean Distance	Minkowski Distance
Time Complexity	$O(n^2t)$	$O(n^2t)$	$O(n^2)$
Scalability	Scalable	Not Scalable	Scalable
Termination Condition	Deterministic	Not Deterministic	Deterministic

Table 2: Comparison between different clustering algorithms that require the number of clusters as a parameter.

Clustering Algorithm	**K-Means**	**Spectral Clustering**	**Agglomerative Clustering**	**BIRCH**
Parameters	Number of clusters	Number of clusters	Number of clusters, Distance	Branching Factor
Metric	Euclidean Distance	Euclidean Distance	Euclidean Distance	Euclidean Distance
Time Complexity	$O(n^2)$	$O(n^3)$	$O(n^2)$	$O(n)$
Scalability	Scalable	Scalable for medium sized samples	Scalable	Scalable
Termination Condition	Deterministic	Deterministic	Deterministic	Deterministic

5.1 Identifying the ideal number of clusters that need to be formed

To determine the ideal value for the number of clusters $\bar{N}$ for those approaches that need this input beforehand, silhouette analysis [45] is employed which is shown in Fig. 3. The silhouette score and silhouette plot are used to measure the distance of separation between clusters. It displays a measure of how close each point in a cluster is to the points in the neighbouring clusters. This measure has a range of [-1, 1] and is a great tool to visually inspect the similarities within clusters and differences across clusters. Silhouette score is calculated using the mean intra-

cluster distance (i)[6], and the mean nearest-cluster distance (n)[7] for each sample. The Silhouette coefficient for a sample is given by

$$\frac{n-i}{max(i,n)}. \tag{26}$$

The typical silhouette plots represent the cluster label on the Y-axis, while the actual silhouette score is on the X-axis. The size/thickness of the silhouettes is also proportional to the number of samples inside that cluster. The higher the silhouette coefficients (the closer to +1), the further away, the cluster's samples are from the neighbouring clusters' samples. A value of 0 indicates that the sample is on or very close to the decision boundary between two neighbouring clusters. Negative values, instead, indicate that those samples might have been assigned to the wrong cluster. Averaging the silhouette coefficients, we can get to a global silhouette score which can be used to describe the entire population's performance with a single value. From Fig. 3, the average silhouette coefficient is found to be high when the number clusters $\bar{N} = 4$ compared to $\bar{N}$=2, 3, or 5 and hence, the ideal number of clusters is identified to be 4. Furthermore, from Fig. 3, we can see that the global silhouette score is slightly over 0.4.

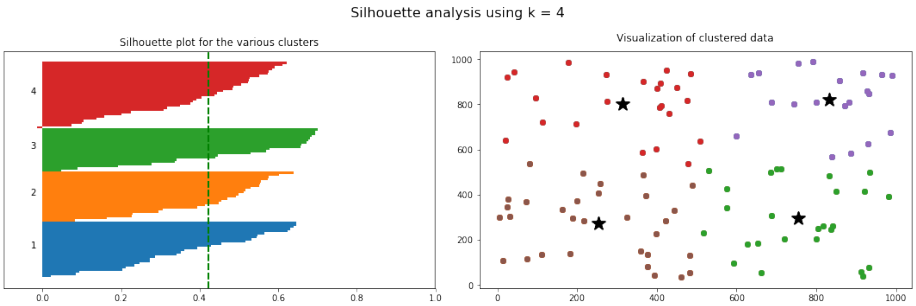


Fig. 3: Silhouette plot of k=4 (best fit) for K-Means Clustering.

5.2 Comparative Evaluation on Clustering Results

Based on the silhouette score[8] estimated for each of the approaches (see Table 3 and Table 4), it is found that K-means provides the best score compared to all other related approaches. Mean-shift, on the other hand, although it comes fifth, has an advantage over K-Means, spectral clustering, Agglomerative clustering, and Birch as it does not need to determine the number of clusters beforehand.

[6] The mean distance within each cluster.

[7] The distance between each sample and the nearest cluster that the sample is not a part of.

[8] This value can be used for: i) identifying the ideal number of clusters that need to be formed by executing a brute force investigation on the approach using with a different number of clusters; and ii) as a metric estimated based on the formula described in Eq. 26 and used for evaluating the approach.

However, on the downside, in mean-shift clustering, the data points are not evenly clustered as can be seen in Fig. 4. Consequently, the K-Means (as shown in [38]) is identified to be most suitable clustering algorithm for large and dynamic data sets.

Table 3: Comparison of various 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 4: 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

Fig. 4: Comparison of the clustering results of K-Means and Mean Shift.

As shown from the results above the best approach (ML algorithm) that should be used in the first stage for clustering our devices into clusters is the K-Means. Thus, this is our first stage Machine learning approach in our solution.

6 Second Stage Approach Selection: Analysis and Comparison of Classification Approaches Investigated

Classification is a supervised process that comprises of a classifier that is trained from pairs consisting of input data and the achievable/desired output value. The classifier should predict the correct class/value based on valid input data. In our work, classification is the second stage of the machine learning engine and aims to categorize the UEs in the clusters to be eligible or non-eligible to become a UE-VBS. UEs are assessed based on the metrics discussed in 4.3.1. Also, it is assumed that for all the eligible UEs and non-eligible UEs, the target variable 'eligibility' is set to 1 and 0, respectively. Additionally, the dataset is split into 70% and 30% of the samples for training and testing, respectively, and the performance of different models is studied and compared based on the performance metrics detailed in the forthcoming subsection 6.2.

In general, this section provides an overview of the classification of the UEs based on their eligibility to become a UE-VBS and explains how different classifiers are trained, evaluated, and compared. Next, it briefly describes the common characteristics of the examined approaches along with a brief description of each approach (i.e., Random Forest, Adaptive Boosting, Gradient Boosting) separately. Some insights on the performance metrics used to evaluate the classification approaches are also provided. Additionally, the problems faced in the classification approach due to inconsistent and imbalanced data from a class imbalance perspective in the dataset and the synthetic minority over-sampling technique (SMOTE) strategy employed to overcome the aforesaid issues and restore equal representation of the two classes (i.e., eligible UEs, non-eligible UEs) are presented. Furthermore, this section presents and explores the classification results obtained post balancing the classes followed by the validation and testing of the investigated classifiers using K-folds, SHAP, and LIME.

6.1 Common characteristics of the investigated approaches

The classifiers (i.e., Random Forest, Adaptive Boosting, Gradient Boosting) compared in our work use decision tree models. Decision trees comprise decision nodes acting as a means to split the data. Each node is a question that helps an individual to arrive at a final decision, denoted by the leaf node. Decision trees seek to find the best split to subset the data, and they are typically trained through the classification and regression tree (CART) algorithm.

Moreover, the investigated approaches are boosting ensemble learning methods and are made up of a set of decision tree classifiers[9] and their predictions are aggregated to identify the most popular result. By combining several individual classification models, the ensemble model [23] tends to be more flexible (less bias)

[9] These models are usually decision trees although other models like Support Vector Machines (SVM) can be employed.

and less data-sensitive (less variance). The two popular ensemble methods are bagging and boosting. A brief explanation of both methods is provided below.

- **Bagging**, also called 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[10].
- **Boosting**: involves training individual models sequentially, with each model learning from the mistake made by the previous model.

Please note that the feature importance calculated by Random Forest for our model is illustrated in Fig. 5.

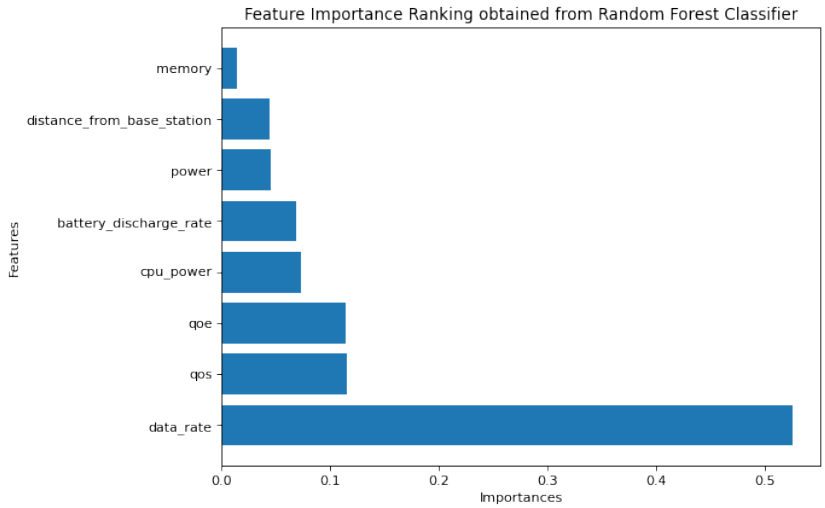


Fig. 5: Feature importance calculated by Random Forest based on the Gini importance.

6.2 Performance Metrics - Confusion Matrix

In our work, we exploit the confusion matrix along with the corresponding scores to quantify the performance of different classifiers. A confusion matrix is a summary of prediction results on a classification problem. The number of correct and incorrect predictions are summarized with count values that are broken down by each class. As shown in Fig. 6, the columns in a confusion matrix represent the true values of the category and the rows represent the predicted values.

- True Eligible (TE): Eligible devices that can become a UE-VBS correctly classified as 'eligible'
- False Eligible (FE): Ineligible devices that cannot become a UE-VBS incorrectly classified as 'eligible'

[10] This means that individual data points can be chosen more than once.

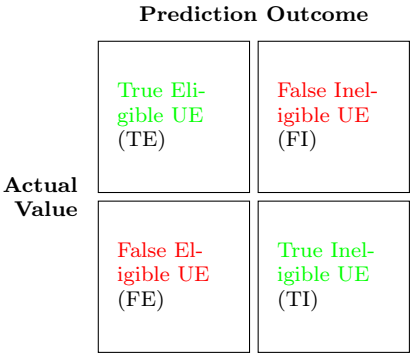


Fig. 6: Confusion Matrix.

– True Ineligible (TI): Ineligible devices that cannot become a UE-VBS correctly classified as 'ineligible'
– False Ineligible (FI): Eligible devices that can become a UE-VBS incorrectly classified as 'ineligible'.

From the confusion matrix, we can compute the following scores to evaluate the performance of the different classifiers:

– Accuracy: It is the most intuitive metric used for model evaluation, describing the number of correct predictions overall predictions.

$$Accuracy = \frac{TE + TI}{TE + TI + FE + FI} \tag{27}$$

– Precision / Specificity : It is a measure of how many of the positive (eligible) predictions made are correct (true), overall the correct (true) and the incorrect (false) positive (eligible) cases in the data.

$$Precision = \frac{TE}{TE + FE} \tag{28}$$

– Recall / Sensitivity / True Eligible Rate (TER): It gives the measure of how many of the positive (eligible) cases the classifier correctly (true) predicted, overall the positive (eligible) cases in the data.

$$Recall = \frac{TE}{TE + FI} \tag{29}$$

– F1-score: It is a measure that combines both precision and recall and is generally used to describe the harmonic mean of the two.

$$F1\ Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{30}$$

– False Ineligible Rate (FIR): It is a measure that indicates what proportion of the negative (ineligible) class got incorrectly classified by the classifier.

$$FIR = \frac{FI}{TI + FI} = 1 - Precision \tag{31}$$

The area under the curve (AUC) - receiver operating characteristics (ROC) (AUC - ROC) plot is another indicator that is used to assess the performance of classification models. Table 5 presents the AUC values and their corresponding inferences. The AUC values are calculated using the trapezoidal approximation in the following Eq. $Y = [0; TER; 1]; X = [0; FIR; 1]; AUC = trapz(Y, X)$. Thus, from the table, we may discern that the higher the area under the curve, the better is the performance of the model. Fig, 7 shows the AUC-ROC analysis for the different classification models trained on our dataset. More specifically, from the Fig. 7, we may discern that the Gradient boosting approach performs marginally better than Random forest and AdaBoost in terms of AUC value, and hence, it can determine the correct positive and the negative class points slightly better than other approaches.. Furthermore, in Table 6, we report the scores obtained from the confusion matrix for the various classification models taken up for investigations in this work. From the table, we may notice that AdaBoost, Gradient Boosting, and Random Forest yield the same accuracy score of 0.95.

Table 5: Interpreting the area under the Receiver Operator Characteristics curve.

AUC Value	**Inference**
$AUC = 0$	The classifier is predicting all negatives as positives, and all positives as negatives
$AUC = 0.5$	The classifier is unable to distinguish the positive and negative class points thereby predicting a random or constant class for all the 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 is able to perfectly distinguish between all the positive and the negative class points correctly

Table 6: Results from the different classification models.

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	0.95

6.3 Class Imbalance & Overcoming Imbalance

In this subsection, we study the strategy adopted to overcome the class imbalance problem. Out of 5250 data points, only 1007 data points account for devices eligible to become a UE-VBS, as can be seen in Fig. 8a. This bias leads to incorrect learning yielding misleadingly optimistic performance called the accuracy paradox. For imbalanced datasets, accuracy is not a reliable metric as it simply captures the

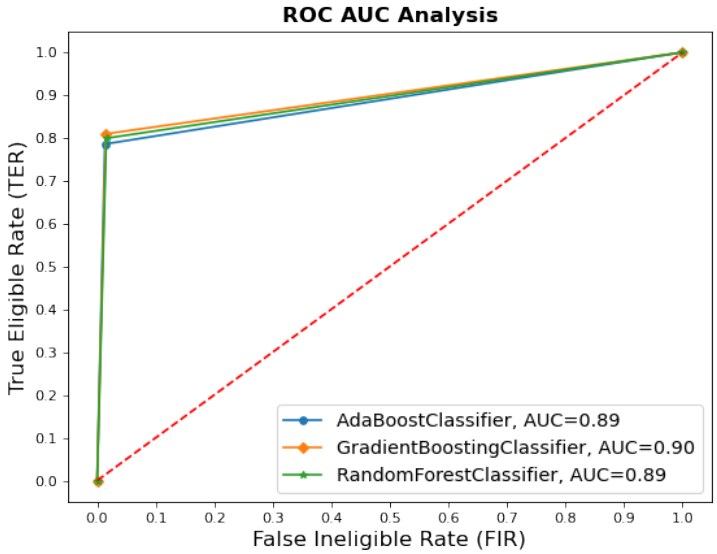


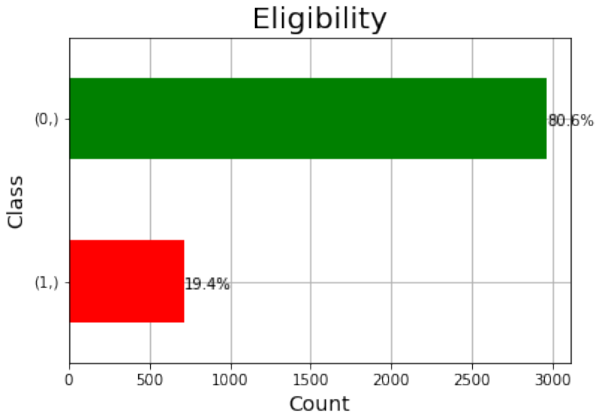
Fig. 7: Area under ROC curve for the different classification models.

proportion of correctly classified instances. In classification problems, the errors made and the target class is usually the area of interest. Therefore, other reliable measures such as precision and recall scores are used to evaluate the performance of the models, and we note that the models obtain higher recall and F1 scores for the negative class (class 0) when compared to the positive class (class 1). The poor performance of the classifiers on class 1 relative to class 0 on the dataset is attributed to the inherent skew towards the class of devices 'ineligible' to become a UE-VBS.

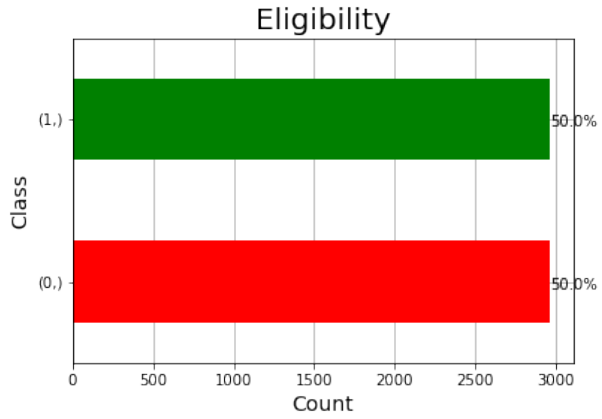
To overcome this bias, we need to perform sampling to ensure equal representation for the two classes. There are two sampling methods to balance the dataset:

- 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.

Synthetic minority over-sampling technique (SMOTE) [34] is an oversampling technique that works by selecting examples that are close in the feature space, thereby, deriving a line between the examples in the feature space and drawing a new sample at a point along that line. SMOTE is employed to balance the training dataset as it synthesizes data points that have smooth variation and high correlation with the existing dataset. Fig. 8b shows the class distribution after employing SMOTE. It can be noticed from the figure that the class imbalance problem is being overwhelmed.



(a) Class distribution before SMOTE



(b) Class distribution after SMOTE

Fig. 8: Overcoming Class Imbalance using SMOTE

6.4 Comparative Evaluation on Classifying Results

As addressed in Section 6.3, SMOTE improves the performance of the classification models. The recall score of class 1 sees a significant increase for the different classifiers as can be witnessed from the results recorded in Table 7 when compared to the results reported in Table 6, which are obtained before applying SMOTE. Also, we may see that both the Gradient boosting classifier and the Random forest classifier achieve the same accuracy of 0.95. Furthermore, from the Table 7, it is seen that the Random forest classifier achieves an F1 score of 0.88 for class 1 ('eligible' UEs), and area 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 in terms of F1 score and ROC value.

Table 7: 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	0.95

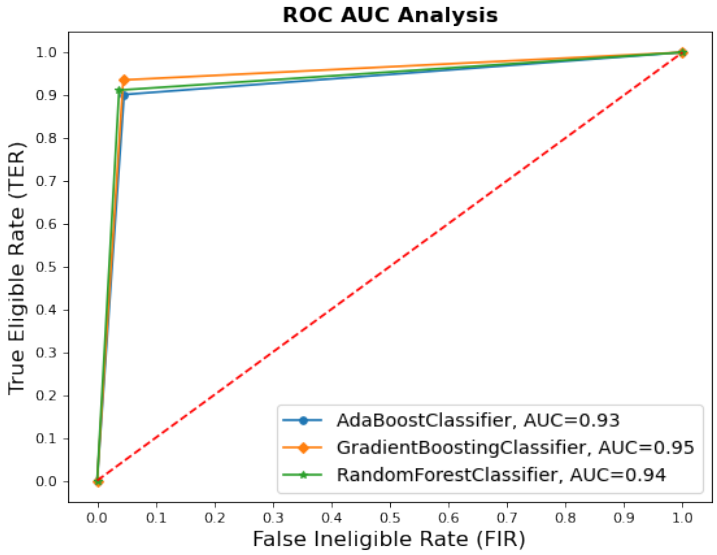


Fig. 9: Area under ROC curve for the different classification models after SMOTE.

6.5 Validation & Testing

This section provides the insights and statistical reasoning behind the selection of the machine learning model for our pipeline and also the Model explainability. Model explainability is a broad concept of analyzing and understanding the results provided by the ML models. It is most often used in the context of "black-box" models, for which it is difficult to demonstrate, how did the model arrive at a specific decision and whether the decision made is technically sound. For the model selection, we tested various classification algorithms with K-Fold testing, followed by SHAP and LIME tests to get a better picture of the model explainability.

6.5.1 K-Folds

K-folds test is a cross-validation test [46]. Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample. The

procedure has a single parameter called Kthat refers to the number of groups that a given data sample is to be split into. The general procedure involves:

1. Shuffle the dataset randomly.
2. Split the dataset into K groups.
3. For each unique group:
   (a) Take a group as the test data.
   (b) Take the remaining groups as training data.
   (c) Fit a model on the training data and evaluate it on the test data.
   (d) Retain the evaluation score and discard the model.
4. Summarize the skill of the model using the model evaluation scores as the sample.

We conducted the test on several algorithms and the results were visualized as a box plot as shown in Fig. 10. Algorithms having higher mean and lower variance of the accuracy score are considered to be efficient models. The outcomes of our test are documented in Table 8. From the results of the test presented in Table 8, 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.

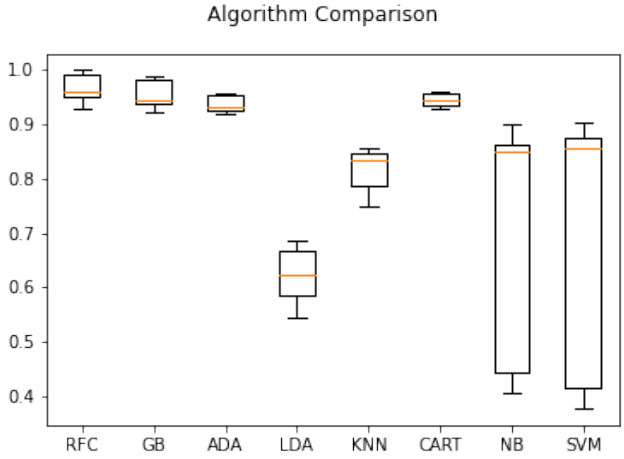


Fig. 10: K-fold cross validation with k=10 for various classification algorithms with accuracy as the performance metric.

6.5.2 SHAP and LIME

SHAP and LIME [20] values are used to explain the interpretability of machine learning models. SHAP and LIME are model agnostic, i.e., model-independent. These values are measured by calculating the impact of the various parameters. LIME values are calculated by local sub-sampling of the dataset, whereas the SHAP values are calculated by removing certain features and calculating their

Table 8: 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

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' which are 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 are influencing f. At the end of the process, for a given sample x, LIME would be able to approximate the prediction of f by determining the individual influence of every feature. Therefore, LIME is able to explain a specific prediction by understanding which features have the most contribution to the prediction. Figs. 11 and 12 illustrate the SHAP and LIME, respectively.

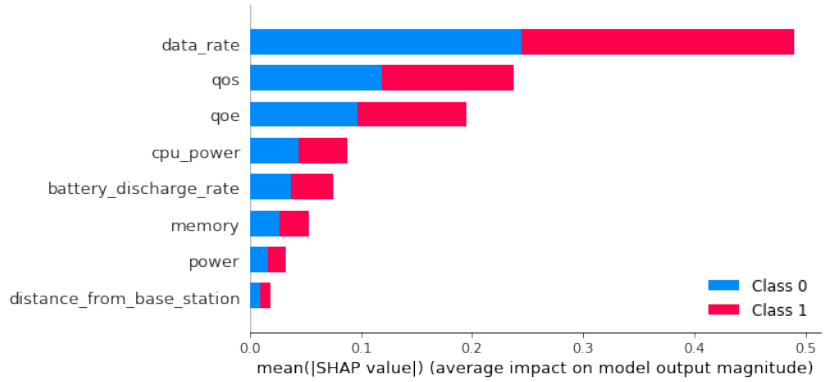


Fig. 11: SHAP.

In this work, SHAP values calculate the importance of a feature by comparing what a model predicts with and without the feature. However, since the order in which a model sees features can affect its predictions, this is done in every possible order, so that the features are fairly compared. Thus, in our work, all

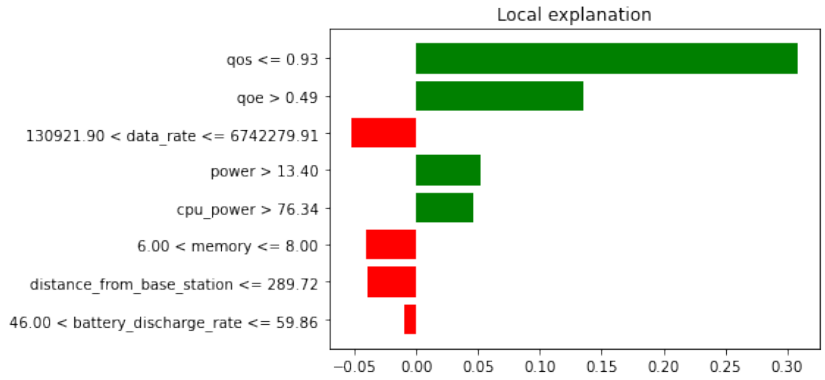


Fig. 12: LIME.

possible coalitions (sets) of feature values are evaluated with and without a selected feature to calculate the exact SHAP value.

The LIME value is used to explain 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 of the parameters. Here, we notice that a UE that has a high data rate and high computational power influence the decision of the model in choosing 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), SHAP value explains the decisions made by the model on a global level. 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 the UEs with higher data rate 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 the selection of a classifier, thereby giving more confidence in the decisions made by the chosen classifier.

Furthermore, in contrast to the study in [28], which is based on the random selection, in our work, we have studied the performance of the models in picking up the UEs that are eligible to become UE-VBSs based on the network parameters.

Table 9: 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

The results of our study is reported in 9. From Table 9, it can be noticed that our model-based selection of the UEs to act as UE-VBSs achieve a five-fold increase in data rate compared to the random selection presented in [28]. Besides, the received power and QoE achieved are considerably higher when our proposed ML model-based selection is invoked than those attained in [28].

7 Heuristic Algorithm for Selecting UE-VBS

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 Alg. 1. The heuristic algorithm selects the UE-VBS from the eligible UEs in a cluster (for generality we named the selected cluster to C). More specifically, the selection process is carried out with the use of 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. This section provides a brief explanation of the algorithm shown in Alg. 1. 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 (D), distance from cluster C's centroid (DC_C) and the thresholds. Then, it creates the following two sets that will be used in the UE-VBs calculation: i) the first set is the $max5DR$ set that has five eligible UEs sorted by the data rate achieved towards BS (DR) with decreasing order; ii) the second set is the $max5DC_C$ set that has five eligible UEs sorted by the distance to the Cluster C's centroid (DC_C) with increasing order. The algorithm then creates the variable $UEVBS$ that will hold the result of the final UE-VBs and assigns to it the first UE in the $max5DC_C$ set that achieves the thresholds as a candidate $UEVBS$. Following this, it iterates over all eligible UEs (as $UE_{eligible}$) in the $max5DC_C$ set and checks each time if the $UE_{eligible}$ is in the $max5DR$ and achieves the thresholds. If yes, then it assigns the $UE_{eligible}$ as $UEVBS$ and the process terminates because the UE-VBS is identified, otherwise, the algorithm continues iterating. Consequently, in the end, the algorithm will return the cluster head and UE-VBS of a cluster or return nothing. In the case of nothing, it implies that the cluster is not created under a BS.

7.1 Evaluation Results and Comparison Based on Sum Rate

This section provides the evaluation results of the proposed heuristic algorithm in terms of sum rate[11] in a mobile network with a different number of devices (i.e., 100, 200, 300). The results of our proposed algorithm are compared with the random approach in which the UE-VBS is selected for each of the clusters from all the clusters' UEs in a random manner and without employing classification.

For our experiment, we generated sample datasets having 100, 200, 300 UEs. At each stage of the experiment (i.e., 100, 200, 300) the generated UEs are clustered with the K-Means clustering algorithm and classified using the Random Forest

[11] The total sum rate is the aggregated Data Rate of all links

Algorithm 1 Selection of UE-VBS Devices as Cluster Heads from the eligible UEs

C: Examined Cluster Number
maxD: maximum distance allowed between the eligible UE device and the BS
minDR: minimum achievable DR allowed between the eligible UE device and the BS
T_C: a set containing information acquired from the clustering
it includes the eligible UEs with their associated Data Rate towards BS (DR), Distance towards BS (D) and Distance from Cluster C Centroid (DC_C)
procedure SELECT_UEVBS($T_C, maxD, maxDR$)
    By considering info included in T_{th} calculate the following:
    $max5DR$: sort by DR with decreasing order and select the top five UEs.
    $max5DC_C$: sort by DC_C with increasing order and select the top five UEs.
    $UEVBS \leftarrow$ first incidence UE from $max5DC_C$ where $D <= maxD \ \wedge \ DR >= minDR$
    $found \leftarrow false$
    while $UE_{eligible} \in max5DC_C \wedge !found$ **do**
        $DR \leftarrow$ DR from $UE_{eligible}$
        $D \leftarrow$ D from $UE_{eligible}$
        if $UE_{eligible} \in max5DR \ \wedge \ D <= maxD \ \wedge \ DR >= minDR$ **then**
            $UEVBS \leftarrow UE_{eligible}$
            $found \leftarrow true$
        end if
    end while
    return $UEVBS$
end procedure

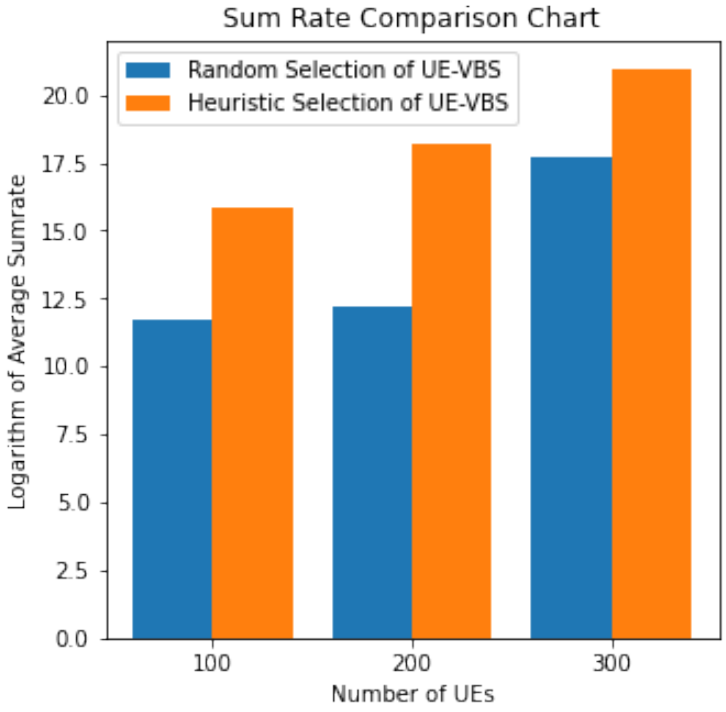


Fig. 13: Comparison of Sum Rates

classifier. From the result of the classification, the eligible UEs are grouped, and from this group, a UE-VBS is selected by using the proposed heuristic algorithm 1 for each of the clusters. The average of the sum rates across each of the clusters is computed for our heuristic model. Similarly, UE-VBSs are chosen randomly and the corresponding sum rates are computed. Since the sum rates are much higher for the heuristic model than the random selection, the graph is very skewed, and hence, the logarithm of the sum rates is considered, which is evinced in Fig. 13. Also, Fig. 13 compares the model-based selection and random selection of UEs to serve as UE-VBSs. From the results, it is clear that our two-stage ML-based selection of UE-VBS results in higher sum rates than random selection.

8 Conclusions And Future Work

In this paper, we delved into the implementation of a two-stage machine learning approach for 5G mobile network augmentation through dynamic selection and activation of UE-VBSs. We have generated and pre-processed the dataset by simulating the UEs geographical coordinates as a non-uniform distribution in the time and space domain. Additionally, another dataset is generated in order to classify the resulting devices of each cluster to eligible UE-VBs and no-eligible UE-VBS. By using the datasets generated, we trained a two-stage machine learning model, followed by a heuristic algorithm for selecting one eligible device per each cluster to become UE-VBS. Moreover, our proposed model has been subjected to statistical testing and analysis using SHAP and LIME. From the results of the performance analysis carried out, we advocate that our proposed ML-aided dynamic selection of a UE-VBS for each of the clusters around the primary BS is efficient as it has resulted in a better achievable data rate than random selection.

As a future direction, the asymmetric distribution of the UEs can be studied. Also, other clustering algorithms like hierarchical clustering, minimum entropy clustering, and X-means can be investigated. Moreover, neural network-based classification algorithms can be studied. Besides, the selection of the cluster head can be implemented using a heuristic algorithm that can use data rate and signal quality based on CQI from the surrounding devices. Additionally, the security aspects related to UE-VBs are also an open issue, because the use of UE-VBs, is likely to increase threat vectors and entry points and thus make it more vulnerable to attacks, such as eavesdropping, IP spoofing, denial of service, and malware attack.

Declarations

8.1 Funding

This work 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.

8.2 Conflicts of interest/Competing interests

There is no potential competing interest

8.3 Ethics approval

Not applicable

8.4 Consent for publication

All authors consent for publication

8.5 Availability of data and materials

The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request after the publication of the article.

8.6 Code availability

The code implemented during the current study is available from the corresponding author on reasonable request after the publication of the article.

8.7 Authors' contributions

All authors are equally contribute in the realisation of this study

References

1. S. Mollahasani, A. Eroğlu, I. Demirkol, E. Onur, Density-aware mobile networks: Opportunities and challenges, Comput. Netw. (2020) 107271.
2. I.F. Akyildiz, A. Kak, E. Khorov, A. Krasilov, A. Kureev, Arbat: A flexible network architecture for qoe-aware communications in 5g systems, Comput. Netw. 147 (2018) 262–279.
3. Small Cell Forum Document, Small cell siting and deployment challenges in hyperdense networks, SFC192, 2017.
4. R. Hou, K. Huang, H. Xie, K.-S. Lui, H. Li, Caching and resource allocation in small cell networks, Comput. Netw. (2020) 107100.
5. C.-C. Lin, C.-T. Tsai, D.-J. Deng, I.-H. Tsai, S.-Y. Jhong, Minimizing electromagnetic pollution and power consumption in green heterogeneous small cell network deployment, Comput. Netw. 129 (2017) 536–547.
6. V. Kumar, S. Yadav, D. Sandeep, S. Dhok, R.K. Barik, H. Dubey, 5g cellular: Concept, research work and enabling technologies, in: Advances in Data and Information Sciences, Springer, 2019, pp. 327–338.
7. I.F. Akyildiz, S. Nie, S.-C. Lin, M. Chandrasekaran, 5G roadmap: 10 key enabling technologies, Comput. Netw. 106 (2016) 17–48.
8. C. Christophorou, A. Pitsillides, I. Akyildiz, Celec framework for reconfigurable small cells as part of 5g ultra-dense networks, in: 2017 IEEE International Conference on Communications (ICC), IEEE, 2017, pp. 1–7.

9. K. Venkateswararao, P. Swain, C. Christophorou, 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), IEEE, 2019, pp. 1–6.
10. A. Gotsis, S. Stefanatos, and A. Alexiou, "Ultradense networks: The new wireless frontier for enabling 5g access," IEEE Vehicular Technology Magazine, vol. 11, no. 2, pp. 71–78, 2016.
11. 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," 2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC), 2018, pp. 1104-1109, doi: 10.1109/IWCMC.2018.8450453.
12. Nisha Panwar, Shantanu Sharma, Awadhesh Kumar Singh,A survey on 5G: The next generation of mobile communication, Physical Communication, vol. 18, part 2, pp. 64-84, 2016, doi:https://doi.org/10.1016/j.phycom.2015.10.006.
13. M. Kamel, W. Hamouda and A. Youssef, "Ultra-Dense Networks: A Survey," in IEEE Communications Surveys & Tutorials, vol. 18, no. 4, pp. 2522-2545, Fourthquarter 2016, doi: 10.1109/COMST.2016.2571730.
14. Y. R. Li, P. Hao, F. Xie, H. Xiao and M. Ren, "Cell and user virtualization for ultra dense network," 2015 IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), 2015, pp. 2359-2363, doi: 10.1109/PIMRC.2015.7343693.
15. D. C. Hogg, "Fun with the Friis free-space transmission formula," in IEEE Antennas and Propagation Magazine, vol. 35, no. 4, pp. 33-35, Aug. 1993, doi: 10.1109/74.229847.
16. S. Wang, W. Guo, Z. Zhou, Y. Wu and X. Chu, "Outage Probability for Multi-Hop D2D Communications With Shortest Path Routing," in IEEE Communications Letters, vol. 19, no. 11, pp. 1997-2000, Nov. 2015, doi: 10.1109/LCOMM.2015.2475428.
17. S. Shalmashi, E. Björnson, M. Kountouris, K. W. Sung and M. Debbah, "Energy efficiency and sum rate when massive MIMO meets device-to-device communication," 2015 IEEE International Conference on Communication Workshop (ICCW), 2015, pp. 627-632, doi: 10.1109/ICCW.2015.7247251.
18. Z. Zhou, M. Dong, K. Ota, J. Wu and T. Sato, "Energy Efficiency and Spectral Efficiency Tradeoff in Device-to-Device (D2D) Communications," in IEEE Wireless Communications Letters, vol. 3, no. 5, pp. 485-488, Oct. 2014, doi: 10.1109/LWC.2014.2337295.
19. Hoßfeld, T., Skorin-Kapov, L., Heegaard, P.E. et al. A new QoE fairness index for QoE management. Qual User Exp 3, 4 (2018). https://doi.org/10.1007/s41233-018-0017-x
20. Scott Lundberg and Su-In Lee, "A Unified Approach to Interpreting Model Predictions", 2017, 1705.07874
21. Ioannou, I., Vassiliou, V., Christophorou, C., & Pitsillides, A. (2020). Distributed artificial intelligence solution for D2D communication in 5G networks. IEEE Systems Journal, 14(3), 4232-4241.
22. 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).
23. Dong, X., Yu, Z., Cao, W. et al. A survey on ensemble learning. Front. Comput. Sci. 14, 241–258 (2020). https://doi.org/10.1007/s11704-019-8208-z
24. Greene, Derek & Cunningham, Padraig & Mayer, Rudolf. (2008). Unsupervised Learning and Clustering. 10.1007/978-3-540-75171-7-3.
25. Gacanin, H. (2018). Autonomous Wireless Systems with Artificial Intelligence. ArXiv, September, 51–59.
26. Fan, Z., Gu, X., Nie, S., & Chen, M. (2018). D2D power control based on supervised and unsupervised learning. 2017 3rd IEEE International Conference on Computer and Communications, ICCC 2017, 2018-Janua, 558–563. https://doi.org/10.1109/CompComm.2017.8322607
27. Ioannou, I., Vassiliou, V., Christophorou, C., & Pitsillides, A. (2020). Distributed Artificial Intelligence Solution for D2D Communication in 5G Networks. IEEE Systems Journal, 14(3), 4232–4241. https://doi.org/10.1109/JSYST.2020.2979044"
28. Jungnickel, V., Manolakis, K., Zirwas, W., Panzner, B., Braun, V., Lossow, M., ... Svensson, T. (2014). The role of small cells, coordinated multipoint, and massive MIMO in 5G. IEEE Communications Magazine, 52(5), 44–51. doi:10.1109/mcom.2014.6815892
29. Mozaffari, M., Saad, W., Bennis, M., & Debbah, M. (2015). Drone Small Cells in the Clouds: Design, Deployment and Performance Analysis. 2015 IEEE Global Communications Conference (GLOBECOM). doi:10.1109/glocom.2015.7417609

30. Prabakar, D., & Saminadan, V. (2020). Improving Spectral Efficiency of Small Cells with Multi-Variant Clustering and Interference Alignment. Journal of Computational and Theoretical Nanoscience, 17(5), 2203–2206. doi:10.1166/jctn.2020.8871
31. Frigui, H. (2008). Clustering: Algorithms and applications. 2008 1st International Workshops on Image Processing Theory, Tools and Applications, IPTA 2008. https://doi.org/10.1109/IPTA.2008.4743793
32. "5G Applications and Use Cases — Digi International." [Online]. Available at:https://www.digi.com/blog/post/5g-applications-and-use-cases Accessed on: 2021-07-24.
33. Rubinstein-Salzedo S. (2018) Big O Notation and Algorithm Efficiency. In: Cryptography. Springer Undergraduate Mathematics Series. Springer, Cham. https://doi.org/10.1007/9783319948188_8
34. Chawla, N., K. Bowyer, Lawrence O. Hall and W. Philip Kegelmeyer. "SMOTE: Synthetic Minority Over-sampling Technique." J. Artif. Intell. Res. 16 (2002): 321-357.
35. Hastie T, Rosset S, Zhu J, Zou H. Multi-class adaboost. Statistics and its Interface, 2009, 2(3): 349–360.
36. Breiman L. Random forests. Machine Learning, 2001, 45(1): 5–32.
37. Friedman J H. Stochastic gradient boosting. Computational Statistics and Data Analysis, 2002, 38(4): 367–378.
38. A. Ahmad and L. Dey, "A k-mean clustering algorithm for mixed numeric and categorical data," Data and Knowledge Engineering, vol. 63, no. 2, pp. 503–527, 2007.
39. Yizong Cheng, "Mean shift, mode seeking, and clustering," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 17, no. 8, pp. 790-799, Aug. 1995, doi: 10.1109/34.400568.
40. Ng, Andrew and Jordan, Michael and Weiss, Yair. (2002). On Spectral Clustering: Analysis and an algorithm. Adv. Neural Inf. Process. Syst. 14.
41. Tian Zhang, Raghu Ramakrishnan, and Miron Livny. 1996. BIRCH: an efficient data clustering method for very large databases. SIGMOD Rec. 25, 2 (June 1996), 103–114. DOI:https://doi.org/10.1145/235968.233324
42. Ankerst, Mihael and Breunig, Markus and Kriegel, Hans-Peter and Sander, Joerg. (1999). OPTICS: Ordering Points to Identify the Clustering Structure. Sigmod Record. 28. 49-60. 10.1145/304182.304187.
43. Frey, Brendan and Dueck, Delbert. (2007). Clustering by Passing Messages Between Data Points. Science (New York, N.Y.). 315. 972-6. 10.1126/science.1136800.
44. M. Pavan and M. Pelillo, "Dominant Sets and Pairwise Clustering," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 1, pp. 167-172, Jan. 2007, doi: 10.1109/TPAMI.2007.250608.
45. Peter, R.J.. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math.. 20.
46. Vanwinckelen, Gitte, and Blockeel, Hendrik. "On Estimating Model Accuracy with Repeated Cross-Validation." BeneLearn 2012: Proceedings of the 21st Belgian-Dutch Conference on Machine Learning, 2012, pp. 39–44.