Buffer Based Self-organizing Fuzzy classifier

1 Introduction

For revolutionary development in technology data is generating through numerous sources as computer simulations, consumer market, different applications, business transactions, autonomous systems, smart phones, satellites, sensors, social media and so on. According to the research, more than 2.5 quintillion bytes of data was generating every day in 2018[28] and this data will be double in every two years has averred by the International Data Corporation (IDC)[38]. All these overwhelming volume and unbounded, structured or unstructured series of data which are arriving continuously are defined as data streams[47]. To extract the preserved knowledge form data stream is quite challenging for data mining or data classification algorithms which have three basic feature: volume, velocity and variety. In recent decades, data classification is popularly studied topic in machine learning[26]. Still now, with the Fuzzy-rule-Based system we have various alternative data classification algorithms as K-Nearest Neighbor(KNN)[41], Baycsian classifiers[29, 21], decision trees[31, 54] , Random forest[40, 48], support vector machines [49, 51] and neural network based classifiers[16, 25] which are widely using in multiple area i.e. image classification[34], remote sensing[52, 53], pattern recognition[19], fault detection[44], user behaviour modeling[30], handwritten digit and face recognition[18, 22, 32, 35], etc.

According to different architectures and operating mechanisms, the existing techniques can be divided into two major part: 1) online[15, 32, 7, 43, 6, 45, 39] and 2) offline[23, 36]. Offline algorithms can extract knowledge from a static dataset and never develop its knowledge based learning architectures, as a result now it is less applicable for Big Data manipulation. On the other hand, online classification techniques (incremental and evolving) can be of one pass type[26].They only store extracted interpretation or information (i.e. rule base[7], key prototype[30], cloud density[10], etc.) in the system from the non stationary data streams meanwhile it abandon previously processed sample[7, 32, 2]. To cope with the overwhelming amount of data in real time, both can address the changing data pattern with updating their system structures and meta-parameters recursively[26]. Those approaches are frequently using in different applications for having its low memory and computational efficiency. But the ordering difference of data sample may affect badly on the performance of online classification algorithms. In part of a real situation, most often some data sample can be available as a static form and the rest can arrive sequentially as streaming data. In that case, learning "from scratch" is inappropriate while the available static data can be train in offline and assumed knowledge can perform on the streaming data[26]. Additionally, online structure can also refute the prior knowledge for non-stationary data by developing its rule base and structure.

In this paper, we propose and describe a data-cloud and fuzzy-rule-based(FRB) classifier grounded at Empirical Fuzzy Sets[4] and recently introduced self-organising fuzzy logic classifier[26]. The algorithm is divided in offline and online training stages. In offline, 0-order AnYa type fuzzy rule is created by extracted knowledge from available static dataset[10]. And in online stage the whole identified rule base of offline stage can be updated continuously by processing streaming data. The online procedure belong "one pass" system and support drift and/or shit in the data pattern. Shortly, the contributions of this study are as follows:

1. SOF[26], the state-of-the-art techniques for classification use a predefined user input variable (granularity level) as threshold and a greedy way for ranking the multimodal density based list can reduce efficacy of the classifier. Here, Prim's algorithm based ranking technique and local average distance of a data-cloud for threshold are using to reduce the dependency on user and increase the efficiency of classifier.
2. A unimodal density[14] and cosine distance based recursive formula is also introduced here for ease of the online stage calculation and to make it free from "curse of dimensionality".

3. From offline/or online stage, some data-cloud may seem irrelevant but can be relevant in future are not considered by existing classifier algorithms. This classifier uses a temporarily memory storage for storing seemingly irrelevant data-clouds. Those data-clouds also be retrieved from temporarily memory storage if it found relevant in future and by functioning some pruning operation it will permanently remove completely irrelevant data-clouds. This operation discard the complexity of creating the new data-clouds and also minimize streaming data processing time.
4. It introduces an efficient and elegant approach to merging existing data-clouds to dominate to dynamic and evolving streaming data.

The overall progress on stream data and data-cloud based classification are concisely but comprehensively discussed in section 3. The proposed methodology and algorithm of fuzzy-rule-based classifier is described in section 4 with proper explanation.

2 Research Question

Considering the earlier described issues, the purpose of this research is to delineate a classifier that can classify stream data more accurately. In this study, the following questions will be answered.

- How to enhance stream data classification accuracy with challenging dataset?
- How to design a stream data classifier which is able more capable to extract information than other existing process?
- Which classifier is better for stream data classification?
- How to use rule base effectively and efficiently by minimizing rule base redundancy?

3 Background Study and Literature Review

3.1 Literature Review

Eclass[13], a prototype/data point potentiality based online classifier is considered as a primitive algorithm in case of stream data classification that creates fuzzy rules with the most potential data point. This process has grounded at two empirical studies, Evolving fuzzy systems from data streams in real-times[12] and Evolving Rule-Based Models[3]. In eclass, antecedent and consequent based fuzzy rule has made using recursive calculation of prototype potentiality and esteemed value of Gaussian-bell type membership function. The overall architecture is one-pass and can develop its rule base "from scratch".

Some problem from the first member[13] of Eclass family has further developed in [7]. In this primitive and empirical study parameter estimation and pruning operation for fuzzy rule base has also introduced with global and local potentiality of a data point. To make the procedure more deterministic it has used cosine dissimilarity to make it free from "curse of dimensionality", where [13] used euclidean type distance for potentiality calculation. But it does not solve the problem of membership functions that generates some esteemed value using aggregation which is significantly differ from real data distribution[10]. On the other hand, it has shown its arresting classification accuracy in numerous benchmark problem data with intrusion detection stream data.

A well-known application of eclass is EVABCD[30], which determines user behavior from some sequence of UNIX command. Using tire based data structure relative frequency has calculated from the command sequence to characterize the user profile. This evolving classifier has the ability to adopt the behavioral changes in command. It has calculated potentiality of a prototype by using the cauchy type function with cosine distance to create a user profile and evolves it considering high/low potential value of an arriving sample data.

All parameterization problems(e.i. centre and spread) of traditional fuzzy rule based system has solved by introducing data-cloud[10, 11] based antecedent part in the rule. This data-cloud concept does not need any membership function and predefined parameter as centre or spread with no shapes or boundaries. In contrast, it can calculate density of a data-cloud

recursively to define antecedent part which represents real data distribution system with a fully data driven process from the data stream. Moreover, this incremental fuzzy rule based classifier is one pass type and does not need to store data samples in the system.

EDA[14] has introduced essential and alternative data analysis operators which is more efficient for both classification[5, 6, 32] and clustering[27]. Data analysis operators as cumulative proximity, standardized eccentricity, density and finally global typicality are non-parametric. Those non-parametric and data driven operators are totally extracted from empirical observation of arriving data samples which does not need any prior unrealistic or restrictive assumptions. Moreover, multimodal typicality based classifier named NaIve EDA classifier has also introduced here which is effective for representing extracted information directly and recursively.

A well-known application for EDA[14] to analyze data in a data distribution system has introduced in [9].Non-parametric and recursive operators eccentricity and typicality has applied for anomaly data detection where higher value of eccentricity indicates anomaly data. Those non-parametric and data driven operators are able to extract information by ensemble data sample properties with recursive calculation which is perfectly applied for streaming data analysis. In addition, Local and global typicality based classifier(Naïve T-EDA Classifier) has proved its domineering performance on numerous alternative classifier[9].

Data cloud and EDA operators has also used parallelly for deep rule based classifiers in Autonomous Learning Multi-Model Systems[6] for stream data classification. In this autonomous learning system, structure has composed with non-parametric data-cloud from extracted knowledge of observed data with recursive calculation of meta-parameters. Anya type fuzzy rule and cauchy function based density also used for rule base identification. Further, this method has also used in [8] for handwritten digits recognition problem which has transparency and high level of interpretability.

Another zero order Anya type fuzzy rule based stream data classifier is TEDAClass[32]. This classifier have used typicality and eccentricity for data analysis where two types of distance(Mahalanobis and Euclidean) have used for recursive calculation. Moreover, it is one-pass classifier where every data sample can proceeded for once and relearn process is not needed while new data sample arrive. This self-adopting and non-parametric system has also introduced merging of AnYa type fuzzy rule for real time classification and to making it computationally efficient.

To enhance performance of fuzzy rule based classifiers ϵFRB system[4] has introduced new form of fuzzy set and fuzzy rule based system. The traditional fuzzy system has parameterization problem with centre, shape and spread for membership function where AnYa type FRB system discard all those limitation by introducing non-parametric and shape free data-cloud. Although data-cloud enhance the objectiveness, it leads to less interpretability and loss of information[4]. As a result, It has combined traditional fuzzy systems and AnYa type fuzzy rule based system to make more efficient system by using unimodal density based membership function.

3.2 Theories and Related Techniques

3.2.1 Data analysis and EDA operators

EDA is an evolving method for data analysis, introduced by Angelov [14]. However, the idea of cumulative proximity, unimodal density and multimodal density has successfully applied in different classification and regression approach[26, 4]. The concepts of cumulative proximity describes the similarity of a n-dimensional data between its surrounding data in a data distribution system and density express the level of closeness of a data point with others. Hence, a data point with low density and high cumulative proximity can be considered as anomaly or isolated data.

Cumulative proximity can be summarize as the sum of square of distance from a particular data sample to all other existing

data sample in the data distribution system[2, 1], as expressed:

$$\pi_K(x_i) = \sum_{j=1}^{K} d^2(x_i, x_j), \quad i = 1, 2, 3....K; \tag{1}$$

Where, x_i/x_j is the data sample of a real time data set or stream X and $X \in R^n$. At K^{th} time instance data space $X^k = \{x_1, x_2, x_3....x_k\}$ and $x_i = [\ x_{i,1}, x_{i,2}, x_{i,3}....x_{i,n}]$, x is the n-dimensional data sample and i denotes the time instance when x_i arrived. The $Unimodal\ density$ of x_i, $i = [1, k]$ has expressed as equation 2:

$$D_k(x_i) = \frac{\sum_{j=1}^{k} \pi_k(x_j)}{2K\pi_k(x_i)} = \frac{\sum_{j=1}^{k}\sum_{l=1}^{k} d^2(x_j, x_l)}{2K\sum_{l=1}^{k} d^2(x_i, x_l)}, \quad i = 1, 2, 3...k; \tag{2}$$

Here, it can be summarized as the sums of the distances from all data samples to all other data samples divided by the sum of the distances of a particular data sample to all other existing data samples of the data space at k^{th} time instance[14, 4]. In real time data distribution system some data samples at k^{th} time instance in X^k can be repeated more than once, literally, $\exists x_i = x_j, i \neq j$. Now, the set of the sorted unique data samples has denoted as $\{U\}_{U_K} = \{u_1, u_2, ..., u_{U_k}\}$, $u_i = [u_{i,1}, u_{i,2}, u_{i,3}....u_{i,n}]$, $\{u\}_{U_k} \subseteq X^k$, $U_k \leq k$, U_k is the number of unique data sample[4, 26]. Also the corresponding frequency of occurrence of a unique data have expressed as $\{f\}_{U_K} = \{f_1, f_2, ..., f_{U_k}\}$ and sum of all frequency ($\sum_{i=1}^{U_k} f_i$) will be k. So, $Multimodal\ density$ of the unique data sample u_i at the time instant k is defined as [9, 5]:

$$D_k^{MM}(u_i) = f_i D_k(u_i) = \frac{f_i \sum_{j=1}^{k} \pi_k(x_j)}{2K\pi_k(x_i)} = \frac{f_i \sum_{j=1}^{k}\sum_{l=1}^{k} d^2(x_j, x_l)}{2K\sum_{l=1}^{k} d^2(u_i, x_l)}, \quad i = 1, 2, 3...k; \tag{3}$$

In equation 1,2 and 3, the function $d^2(x_j, x_l)$ or $d^2(u_i, x_l)$ denotes the square of distance between two data point. Both $Unimodal\ density$ and $Multimodal\ density$ using those equation can be calculated in offline for static dataset. Different process for evolving or stream data has developed using Mahalanobis distance[32] and Euclidean distance[6].

$$d_{cos}(a_k, b_l) = 1 - \frac{\sum_{j=1}^{n} a_{kj} b_{lj}}{\sqrt{\sum_{j=1}^{n} a_{kj}^2 \sum_{j=1}^{n} b_{lj}^2}}, \quad a \text{ and } b \text{ are } n \text{ dimensional data;} \tag{4}$$

This study has introducing a cosine distance based formulae for ease of efficient online calculation and to make this system free from "Curse of dimensionality". Let us assume that the distance type is cosine distance (4), and then derive some formulae to describe the process of recursive calculation of $Unimodal\ density$:

$$\begin{aligned} D_k(x_i) &= \frac{\sum_{j=1}^{k}\sum_{l=1}^{k} d^2(x_j, x_l)}{2K\sum_{l=1}^{k} d^2(x_i, x_l)}, \quad i = 1, 2, 3...k; \\ &= \frac{\sum_{j=1}^{k}\sum_{l=1}^{k} d^2(x_j, x_l)}{2K\sum_{l=1}^{k}\left(1 - \frac{\sum_{j=1}^{n} x_{ij}x_{lj}}{\sqrt{\sum_{j=1}^{n} x_{ij}^2 \sum_{j=1}^{n} x_{lj}^2}}\right)^2} \end{aligned} \tag{5}$$

Now, by considering the denominator part and simplifying it we found:

$$
\begin{aligned}
2K\sum_{l=1}^{k}\left(1-\frac{\sum_{j=1}^{n}x_{ij}x_{lj}}{\sqrt{\sum_{j=1}^{n}x_{ij}^2\sum_{j=1}^{n}x_{lj}^2}}\right)^2 &= 2K\sum_{l=1}^{k}\left(1-\frac{\sum_{j=1}^{n}x_{ij}x_{lj}}{\sqrt{\sum_{j=1}^{n}x_{ij}^2}\sqrt{\sum_{j=1}^{n}x_{lj}^2}}\right)^2 \\
&= 2K\sum_{l=1}^{k}\left(1-\frac{f_l}{hg_l}\right)^2; \quad \text{Here, } f_l=\sum_{j=1}^{n}x_{ij}x_{lj},\ \ h=\sqrt{\sum_{j=1}^{n}x_{ij}^2},\ \ g_l=\sqrt{\sum_{j=1}^{n}x_{lj}^2}; \\
&= 2K\sum_{l=1}^{k}\left(1-\frac{2f_l}{hg_l}+\left[\frac{f_l}{hg_l}\right]^2\right) \\
&= 2K\sum_{l=1}^{k}\left(1-\frac{2f_l}{hg_l}+\left[\frac{f_l^2}{h^2g_l^2}\right]\right) \\
&= 2K\left(\sum_{l=1}^{k}(1)-\sum_{l=1}^{k}\frac{2f_l}{hg_l}+\sum_{l=1}^{k}\left[\frac{f_l^2}{h^2g_l^2}\right]\right) \\
&= 2K\left(k-\frac{1}{h}\left(\sum_{l=1}^{k}\frac{2f_l}{g_l}+\frac{1}{h}\sum_{l=1}^{k}\left[\frac{f_l^2}{g_l^2}\right]\right)\right) \\
&= 2K\left(k-\frac{1}{h}\left(2B_k+\frac{1}{h}D_k\right)\right)
\end{aligned}
\tag{6}
$$

Where $B_k=\sum_{l=1}^{k}\frac{f_l}{g_l}$ and $D_k=\sum_{l=1}^{k}\frac{f_l^2}{g_l^2}$. By observing the variable f_l and g_l, we can assume that they depend on all the data sample represent by l but variable h only depends on the arriving x_i data attribute. In addition, both B_k and D_k can be calculated recursively also introduced in [30] as:

$$
B_k=\sum_{j=1}^{n}x_{ij}b_{kj};\quad b_{kj}=b_{(k-1)j}+\sqrt{\frac{x_{kj}}{\sum_{l=1}^{n}(x_{kl})^2}};\quad b_{1j}=\sqrt{\frac{x_{kj}}{\sum_{l=1}^{n}(x_{kl})^2}} \tag{7}
$$

and

$$
D_k=\sum_{j=1}^{n}x_{kj}\sum_{p=1}^{n}x_{kp}d_{kj}^{p};\quad d_{kj}^{p}=d_{(k-1)j}^{p}+\frac{x_{kj}x_{kp}}{\sum_{l=1}^{n}(x_k^l)^2};\quad d_{1j}^{1}=\frac{x_{1j}x_{11}}{\sum_{l=1}^{n}(x_1^l)^2} \tag{8}
$$

Finally, we get recursive function of $Unimodal\ \ density$ as:

$$
\begin{aligned}
D_k(x_i) &= \frac{\sum_{j=1}^{k}\sum_{l=1}^{k}d^2(x_j,x_l)}{2K\sum_{l=1}^{k}\left(1-\frac{\sum_{j=1}^{n}x_{ij}x_{lj}}{\sqrt{\sum_{j=1}^{n}x_{ij}^2\sum_{j=1}^{n}x_{lj}^2}}\right)^2} \\
&= \frac{\sum_{j=1}^{k}\sum_{l=1}^{k}d^2(x_j,x_l)}{2K\left(k-\frac{1}{h}\left(2B_k+\frac{1}{h}D_k\right)\right)} \\
&= \frac{\sigma_{k-1}+2\gamma_k}{2K\left(k-\frac{1}{h}\left(2B_k+\frac{1}{h}D_k\right)\right)}
\end{aligned}
\tag{9}
$$

where, the distance between k^{th} data to all other data in the data space, $\gamma_k=k-\frac{1}{h}\left(2B_k+\frac{1}{h}D_k\right)$. The distance between all pair for the streaming data, $\sigma_k=\sigma_{k-1}+2\gamma_k$ and σ_1 for online training stage will be the sum of distance of all pair static available data in offline stage.

In addition, for online process we have to calculate the density of existing data cloud. We can use the variable and density calculated in $(k-1)^{th}$ time instance as:

$$
\begin{aligned}
D_{k-1}(cloud^*) &= \frac{\sum_{j=1}^{k-1}\sum_{l=1}^{k-1} d^2(x_j, x_l)}{2(K-1)\sum_{l=1}^{k-1} d^2(cloud^*, x_l)} \\
D_{k-1}(cloud^*) &= \frac{\sigma_{k-1}}{2(K-1)\sum_{l=1}^{k-1} d^2(cloud^*, x_l)} \\
\sum_{l=1}^{k-1} d^2(cloud^*, x_l) &= \frac{\sigma_{k-1}}{2(K-1)D_{k-1}(cloud^*)}
\end{aligned}
\tag{10}
$$

Further using formulae 10, we will calculate the density of existing data-cloud at k^{th} time instance as:

$$
\begin{aligned}
D_k(cloud^*) &= \frac{\sum_{j=1}^{k}\sum_{l=1}^{k} d^2(x_j, x_l)}{2K\sum_{l=1}^{k} d^2(cloud^*, x_l)} \\
&= \frac{\sigma_{k-1} + 2\gamma_k}{2K\left[\sum_{l=1}^{k-1} d^2(cloud^*, x_l) + d^2(cloud^*, x_k)\right]} \\
&= \frac{\sigma_k}{2K\left[\frac{\sigma_{k-1}}{2(K-1)D_{k-1}(cloud^*)} + d^2(cloud^*, x_k)\right]}
\end{aligned}
\tag{11}
$$

This section already described about EDA operators and the newly introduced recursive formulae for density calculation. Section 3.2.2 will discuss about Data-cloud and AnYa type fuzzy rule.

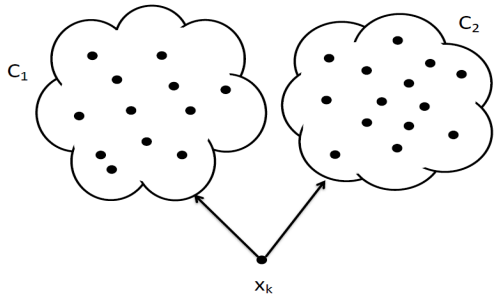

Figure 1: *Insertion of x_k in a data-cloud*

3.2.2 Data-cloud and AnYa type fuzzy rule

The data-cloud introduced in [10] and further successfully applied in numerous application[] has made a new era for data classification[] or data clustering[]. Data-cloud can identified through some set of data with common properties of cluster but it does not have any predefined boundaries or a particular shape. Comparing with traditional membership function(i.e. trapezoidal, triangular and gaussian), data-cloud has much more objective representation of real data distribution.

Equation (9) can calculate density of arriving data sample x_k independently in online training stage to identify in which cloud it belong as shown in figure 1. In this figure, the proposed system can determine the membership of arrived data x_k between two data-cloud c_1 and c_2 based on calculated density and proposed threshold in equation (threshold using avg-dist). Holding that condition proposed classifier will determine the membership of arrived data x_k in which data-cloud it will significantly affect or does it need a new data-cloud.

Since, this algorithm is recursive, it will not store past data in online training stage. Instead, some extracted information from data through variable are needed for every data-cloud: density of centre ($Cden_i^k$), number of samples (S_i^k), average distance between centre to each sample (Ad_i^k) and age of a data-cloud (Age_i^k) at k^{th} time instant.

For example, figure 2 depicts a scenario (two feature of multiple feature dataset analysis of class 1) for cloud c_1 and c_2 after arrival of k^{th} data. The density of each data-cloud $Cden_1^k = .99$ and $Cden_2^k = .86$, the number of samples belong to each

data-cloud S_1^k = 8 and S_2^k = 6, average distance Ad_1^k = .02 and Ad_2^k = .039 and the age of each data-cloud Age_1^k = 89.04 and Age_2^k = 54.81.

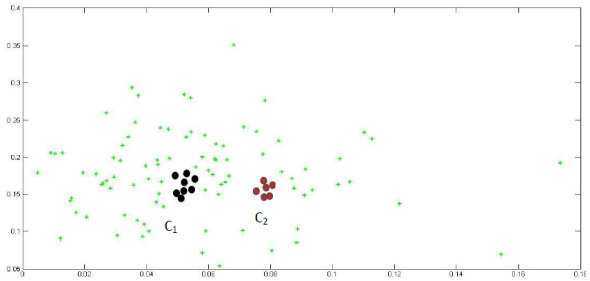

Figure 2: *Data-cloud of samples (Multiple feature dataset, two of the features).*

The alternative approach of widely used fuzzy rule based system Takagi-Sugeno[50] or Mamdani[37] type is AnYa type FRB system which was introduced in [10]. The traditional Mamdani approach is more linguistic than TS system but they provide same type of antecedent part, one is fuzzy-set-based and other is function based whereas AnYa type fuzzy rules are more objective, non-parametric and compact.It does not need any predefined membership function. AnYa type fuzzy rule is as:

$$\textbf{IF } (x \sim C_1^j) \textbf{ OR } (x \sim C_2^j) \textbf{ OR } (x \sim C_3^j) \textbf{ OR } ..... \textbf{ OR } (x \sim C_N^j) \textbf{ THEN } Class_i \longleftarrow x \tag{12}$$

where C_i^j denotes j^{th} data-cloud of i^{th} class for n-dimensional input data vector, x =[x_i, x_2, x_3...x_n]; N is the number of class, j is the number of data-cloud extracted from the data samples which can varied class to class and ' ' expressed the similarity between x and j^{th} data-cloud. The class label of the arrived data can esteemed by winner-takes-all strategy which is perfectly described in section 4.

4 Proposed Methodology of Classifier

In this section, the offline training architecture, online training architecture and decision-making processes of the proposed classifier are depicted in detail. In addition, the offline training will be processed with the static dataset and variables will be prepared per class for using in online training stage. In online training phase all calculation will be conducted at k^{th} time instance.

4.1 Offline training architecture

This proposed classifier will determine the superior data points from each class uniquely and form an AnYa type fuzzy rule (as equation 12) based on data points which is determined from every class. Data points identified from each class will not be affect by each other. As mentioned earlier in section 3.2.1, all EDA operators will be operated in k^{th} time instance for C^{th} class of data. So, the set of unique data sample will be $\{u\}_{U_K^c}^c = \{u_1^c, u_2^c, ..., u_{U_k^c}\}$, and their frequencies, $\{f\}_{U_K^c}^c = \{f_1^c, f_2^c, ..., f_{U_k^c}\}$. The number of unique data sample of c^{th} class is U_k^c and $\sum_{c=1}^{C} U_k^c$ is equal to the total data samples of static dataset.

Firstly, the proposed classifier will determine superior data points from the dataset by using multimodal density formulae (3) for per class re-written as:

$$D_{k^c}^{MM}(u_i^c) = \frac{f_i^c \sum_{j=1}^{k^c} \sum_{l=1}^{k^c} d^2(x_j^c, x_l^c)}{2K^c \sum_{l=1}^{k^c} d^2(u_i^c, x_l^c)}, \quad i = 1, 2, 3...U_k^c; \tag{13}$$

where, K^c is denoted by the number of data sample of class C. Now, ranking system will be conducted by using prim's algorithms concept, esteemed density values and the distance between every pair unique value.

To preparing the tree, maximum density ($\arg\max_{i=1}^{U_k^c}(D_{k^c}^{MM}(u_i^C))$) of a unique data sample will be the root node and the nearest ($\arg\min_{i=1}^{U_k^c}(d(root, u_i^C))$) data sample of root will be the first ancestor. Then the nearest data sample of tree nodes will be added next as ancestor node. The newly added node can be nearest of root or the first node. So, from which node it is nearest will be the parent node of that ancestor and one unique data sample will not be selected for tree more than one time. Repeating this process and by using all unique data the tree will be generated. An illustration has made using 100 data of class 2 from occupancy detection dataset in figure 3. The node number denotes serial no of an unique sample data in the static dataset and node-61 is root. The sign "⟵" indicates a child node of a parent node and the parent node containing more than one child (out-degree in the fig. 3) will be considered as junction parent node in this study.

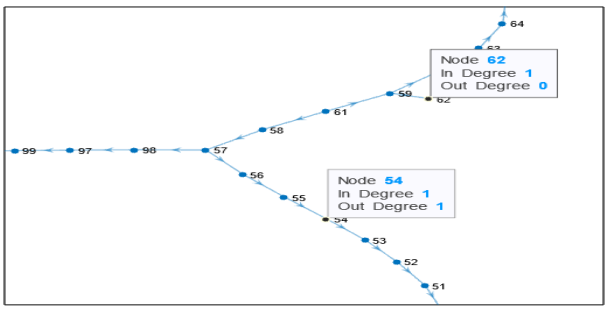

Figure 3: *Illustration of a tree generated from 100 data of class 2 from occupancy detection dataset.*

Condition 1:

$$\textbf{IF } (D_c^{MM}(node_i) > D_c^{MM}(node_{i+1})) \textbf{ AND } (D_c^{MM}(node_i) > D_c^{MM}(node_{i-1})) \textbf{ THEN } node_i \in Cloud\_centre_c \tag{14}$$

From the tree, three types of path segment will be used to identify the superior data points or prototypes: 1) root to a node before junction parent node, 2) junction parent node to a node before another junction parent node and 3) junction parent node to leaf node. If one junction parent node have more than one valid path, then the first path will be chosen by considering highest density. Applying condition **1** and three path segment of tree, the identified unique data points or prototypes will be assigned in $Cloud\_centre_c$ for C^{th} class as a data-cloud centre.

Before cloud merging in offline mode, firstly we have to make data-clouds using all the sample data available in offline mode and by using widely used Voronoi tessellation[17] as:

$$\textbf{winning Cloud\_centre } = \arg\min_{i=1}^{m}(d(x_j, Cloud\_centre_c^m)); \;\; x_j \in X^c \tag{15}$$

The cloud centre will always grab its nearby data samples to make an effective data-cloud and Average distance between cloud centre to its sample, $Ad_i^c = \sum d(Cloud\_centre_c^i, x_j)$ / no of sample in i^{th} data-cloud. If any cloud exits into the potential area (distance between two cloud less than equal ones Ad_i^c) of another cloud then according to condition 2 (16) those clouds will be merged (multiple data-cloud to one data-cloud) into one data-cloud, the i^{th} one. The offline merging procedure has delineates in figure 4 where seemingly elliptical structure for data-cloud has used for simplicity (data-cloud does not have any shapes or boundaries). In figure 4(a), the centre of data-cloud$_2$ is located on\inside data-cloud$_1$ and data-cloud$_3$ is located outside of the potential zone of data-cloud$_1$. Hence, the merging result of data-cloud 1 and 2 has depicted in figure 4(b). In contrast, no merge operation takes place between data-cloud 1 and 3.

Condition 2:

$$\textbf{IF } \;\; \exists\, (d(Cloud\_centre_c^i, Cloud\_centre_c^{j*}) \textbf{ AND } i \neq j) \leq Ad_i^c \;\; \textbf{THEN } \forall (Cloud\_centre_c^{j*}) \in Cloud\_centre_c^i \tag{16}$$

If **Condition 2** is satisfied then some of our variables or parameter (no of sample in a data-cloud, data-cloud centre and average distance) for i^{th} data-cloud will be updated as follows:

$$
\begin{aligned}
S^c_{Cloud\_centre^i_c} &= S^c_{Cloud\_centre^i_c} + S^c_{Cloud\_centre^{j*}_c} \\
centre^c_i &= \frac{\sum X^c_i}{S^c_{Cloud\_centre^i_c}} \\
Ad^c_i &= \frac{\sum d(centre^c_i, X^c_i)}{S^c_{Cloud\_centre^i_c}}
\end{aligned} \tag{17}
$$

Where, $S^c_{Cloud\_centre^i_c}$ denotes number of data samples in cloud i of class c and $S^c_{Cloud\_centre^{j*}_c}$ denotes total number of sample of all j^{th} cloud in class c which satisfied **condition 2**. Updated centre and average distance of i^{th} cloud is $centre^c_i$ and Ad^c_i. After the variables or parameter of all classes are updated, the offline training stage will prepare AnYa type fuzzy rule base as 18, where N and c indicates number of cloud and number of class.

$$
\textbf{IF } (x \sim Cloud\_centre^c_1) \textbf{ OR } (x \sim Cloud\_centre^c_2) \textbf{ OR } \ldots\ldots \textbf{ OR } (x \sim Cloud\_centre^c_N) \textbf{ THEN } Class_c \longleftarrow x \tag{18}
$$

Before the ending of offline training stage it will calculate the parameter σ^c, γ^c and unimodal density of identified data-cloud ($D(Cloud\_centre^c_i)$) per class for recursive calculation of online training stage. The **Algorithm 1** has illustrated overall procedure of offline training architecture.

Algorithm 1: Offline Training Architecture

Input: Static dataset.

Output: Data-cloud, Average distance of each data-cloud (Ad^c_{cloud}), data-cloud centre ($centre^c_{cloud}$), support of each data-coud (S^c_{cloud}), σ^c and γ^c .

Step 1: Find out unique data samples.

Step 2: Calculate multimodal density ($D^{MM}_c(u^c_i)$) for each unique data sample .

Step 3: Prepare tree to rank the unique data sample using prim's algorithm.

Step 4: Apply **Condition 1** in the tree to find out superior unique data sample.

Step 5: Prepare data-cloud using formulae 15.

Step 6:

for *all the existing data-cloud apply* ***Condition 2*** **do**

  if ***Condition 2*** *is satisfied* **then**

    Update i^{th} data-cloud parameters (Ad^c_{cloud}, S^c_{cloud} and $centre^c_{cloud}$) using formulae 17.

    Delete all $j^{th} data - cloud$.

  end

end

Step 7: Create AnYa type data-cloud based fuzzy rule using 18.

Step 8: Calculate per class parameter σ^c and γ^c.

4.2 Evolving training architecture

In this subsection, the architecture of evolving training and removing irrelevant data-cloud is detailed as follows. As this online approach is "one-pass", it will not store previous data samples rather it will update its data-cloud and per class variable or parameters obtained from offline training stage. Irrelevant data-cloud will be stored in temporary memory storage and within some conditions it will be retrieved or removed permanently. Data-cloud will also be merged when conditions will be satisfied to make this approach more computation and memory efficient. Before the beginning of evolving train process age of data-cloud(Age^c_0) will be initialized as zero for each data-cloud of each class. Further, considering this age of a particular data-cloud we will identify a data-cloud as relevant or irrelevant. Moreover, in the following process we have to assume that the online training process is executing on a sample data belongs c^{th} class.

For each arriving data sample the system calculate its unimodal density using recursive formulae (9) with variables σ^c_k and

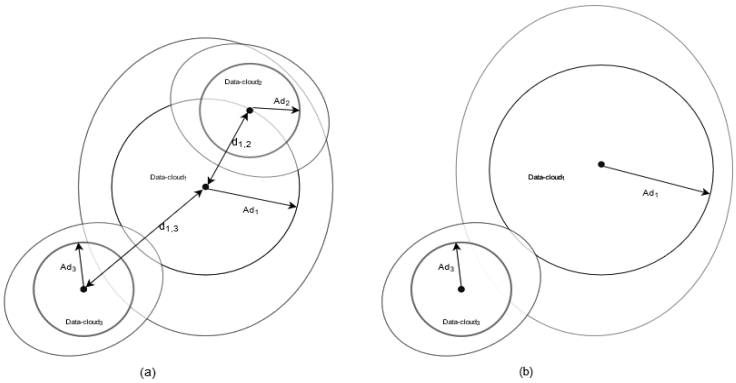

Figure 4: *Illustrates merging procedure of data-cloud in offline mode, where fig. (a) describes the state before cloud merging and fig. (b) describes the state after merging data-cloud 1 with 2.*

γ_k^c. At the same time we have to update existing data-cloud density recursively using density of $(k-1)^{th}$ time instance as early mentioned formulae 11.

Condition 3:

$$\begin{aligned} \textbf{IF } (D_k^c(x_k^c)) > \max(D_k^c(Cloud\_centre^*)) \textbf{ OR } (D_k^c(x_k^c)) < \min(D_k^c(Cloud\_centre^*)) \\ \textbf{OR } (d(x_k^c, Cloud\_centre^*) > \max(Ad_k^c)) \textbf{ THEN } x_k^c \in Cloud\_centre^c \end{aligned} \tag{19}$$

Condition 3 will be satisfied when the newly arrived k^{th} data will contain maximum or minimum density than any existing data cloud and the distance between k^{th} data x_k^c with any existing data-cloud is greater than maximum average distance of all data-cloud, new data-cloud will be created with early mentioned variables or parameters as follows:

$$Cloud\_centre_{new}^c = x_k^c; \quad Ad_{new}^c = 0; \quad Age_{new}^c = 0; \quad S_{new}^c = 1; \quad D_{new}^c = D_k^c(x_k^c) \tag{20}$$

The condition will not be satisfied when it is nearest of a data-cloud and identified nearest data-cloud, $id = \arg\min(d(x_k^c, Cloud\_centre_*^c))$. The parameters or variables of identified data-cloud will be updated as follows:

$$\begin{aligned} Cloud\_centre_{id}^c &= \frac{(Cloud\_centre_{id}^c * S_{id}^c) + x_k^c}{S_{id}^c + 1}; \\ Ad_{id}^c &= \frac{(Ad_{id}^c * S_{id}^c) + d(x_k^c, Cloud\_centre_*^c)}{S_{id}^c + 1}; \\ sum\_of\_k_{cloud,k}^c &= sum\_of\_k_{cloud,k-1}^c + k; \quad S_{id}^c = S_{id}^c + 1; \end{aligned} \tag{21}$$

where sum_of_k denotes the sum of time index when k^{th} data has arrived in a specific data-cloud and it is needed for recursively age calculation of a data-cloud.

Now, we will calculate age of a data-cloud recursively to find out irrelevant data-cloud as follows:

$$Age_{cloud,k}^c = k - \frac{sum\_of\_k_{cloud,k}^c}{S_{cloud,k}^c} \tag{22}$$

If data sample is assigned to a existing data-cloud, its age will be smaller and if no data sample is assigned to existing data-cloud, its age will be added by one[7]. In addition, age of a data-cloud will always follow the range [0,k]. The following simple conditions will determine which data-cloud will be stored in temporary memory storage as a seemingly irrelevant data-cloud.

Condition 4:

$$\textbf{IF } (Age_{i,k}^c) > \overline{Age_k^c} + std(Age_k^c) \textbf{ THEN } Cloud\_centre_{i,k}^c \in temporary\_storage \textbf{ AND } survival\_age_{i,k}^c = 0.5 \tag{23}$$

where i is all the data-cloud of a specific class c, $\overline{Age_k^c}$ denotes the mean age and $std(Age_k^c)$ is the standard deviation of the age. According to the condition a data-cloud will be considered irrelevant when its age is more than the "one-sigma" value. As mentioned earlier this system will retrieve data-cloud from the temporary memory storage and remove its survival_age when the following conditions will be satisfied.

Condition 5:

$$\textbf{IF} \quad (Age_{i,k}^c) < \overline{Age_k^c} + std(Age_k^c) \ \textbf{THEN} \ Retrieve \longleftarrow Cloud\_centre_{i,k}^c \ \textbf{AND} \ Remove \longleftarrow survival\_age_{i,k}^c \tag{24}$$

According to the condition, data-cloud will be retrieved from the temporary memory storage when it will be found as relevant to the current data stream.

For every iteration, the survival_age of a data-cloud stored in temporary memory storage will be reduced by $\frac{1}{1000}$ as decay. The process can identify dying data-cloud which are completely irrelevant to the recent data flow or stream for a long time. A died data-cloud will be identified when its survival_age($survival\_age_{i,k}^c$) will be less then or equal to zero and will be removed permanently from the memory storage. The process of identifying and removing data-cloud is as follows:

Condition 6:

$$\textbf{IF} \quad survival\_age_{i,k}^c \leq 0 \ \textbf{THEN} \ Remove \longleftarrow Cloud\_centre_{i,k}^c \tag{25}$$

Furthermore, from **condition 6**, the irrelevant data-cloud with non-positive survival_age are identified and are completely removed from the memory storage. The data-cloud with zero or negative survival_age can be identified as anomaly data. However, this study is out of that concept.

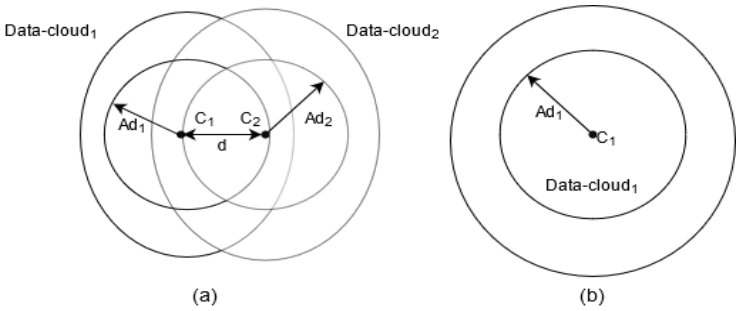

Figure 5: *Merging procedure of data-cloud in online mode, where fig. (a) describes the state before data-cloud merging and fig. (b) describes the state after merging data-cloud 1 with 2 according to* ***condition 7****.*

In order to make number of data-clouds limited, at the same time, extract the dynamic or evolving characteristics of data, our proposed system is able to merge two overlapping data-clouds. A data-cloud merging operation will be executed when distance between two nearest data-cloud is significantly low. The condition for this fully autonomous process is as follows:

Condition 7:

$$\begin{aligned} \textbf{IF} \quad d(Cloud\_centre_{i,k}^c, Cloud\_centre_{j,k}^c) < \frac{Ad_{i,k}^c}{2} + \frac{Ad_{j,k}^c}{2} \ \textbf{THEN} \ & Cloud\_centre_{j,k}^c \in Cloud\_centre_{i,k}^c; \\ & j = \arg\min_{j=1}^{N^c} d(Cloud\_centre_{i,k}^c, Cloud\_centre_{j*,k}^c); \end{aligned} \tag{26}$$

In summary, the condition depicts that two clouds will be merged only when distance between i^{th} data-cloud and j^{th} data-cloud is less than the sum of their half of local average distance and the j^{th} data-cloud will be selected on basis of minimum distance from i^{th} data-cloud. Figure 5 describes the procedure of online data-cloud merging. In fig. 5(a), both data-cloud satisfied **condition 7** and the resulting merged data-cloud has shown in fig. 5(b).

This merging process will control uncontrolled growth of data-clouds to make this system more dynamic. As, two data-cloud

has merged, we have to update its parameters or variables as follows:

$$\begin{aligned} Cloud\_centre_i^c &= \frac{(Cloud\_centre_i^c * S_i^c) + (Cloud\_centre_j^c * S_j^c)}{S_i^c + S_j^c}; \\ Ad_i^c &= \frac{(Ad_i^c * S_i^c) + (Ad_j^c * S_j^c)}{S_i^c + S_j^c}; \\ Age_i^c &= k - \frac{(sum\_of\_k_i^c * S_i^c) + (sum\_of\_k_j^c * S_j^c)}{S_i^c + S_i^c}; \\ sum\_of\_k_i^c &= sum\_of\_k_i^c + sum\_of\_k_j^c; \\ S_i^c &= S_i^c + S_j^c; \end{aligned} \tag{27}$$

After online merging operation of data-cloud it will update its AnYa type fuzzy rule base according to 18 and will read next data sample or be shifted at validation stage.

Algorithm 2: Online Training Architecture

Input: Data sample x_k^c as stream data and parameters (Ad_{cloud}^c, $centre_{cloud}^c$, S_{cloud}^c, σ^c and γ^c) from offline training stage.

Output: AnYa type data-cloud based fuzzy rule.

while *Data stream not ends* **do**
- Initialize Age_{ik}^c, $sum\_of\_k_{ik}^c$, B_k^c and D_k^c .
- Calculate unimodal density of arrived data $D_k^c(x_k^c)$ applying formulae 9.
- Calculate density of existing data-clouds $D_k^c(Cloud\_centre^*)$ using recursive formulae 11.
- **if** ***Condition 3*** *is satisfied* **then**
  - Create new data-cloud with parameter of 20.
- **else**
  - Enter in a existing data-cloud and update parameter of 21.
- **end**
- **for** *Every existing data-cloud* **do**
  - Update age (Age_{ik}^c) recursively using formulae 22.
- **end**
- **for** *Every existing data-cloud* **do**
  - **if** ***Condition 4*** *is satisfied* **then**
    - Move data-cloud into temporary_storage (Buffer).
    - Assign survival age of the data-cloud, $survival\_age_{i,k}^c$ = 0.5.
  - **end**
- **end**
- **for** *Every data-cloud in temporary_storage (Buffer)* **do**
  - Reduce decay from survival age as $survival\_age_{i,k}^c$ = $survival\_age_{i,k}^c$ - decay.
  - **if** ***Condition 5*** *is satisfied* **then**
    - Retrieve data-cloud from temporary_storage (Buffer).
    - Remove survival age of the data-cloud.
  - **end**
- **end**
- **for** *Every data-cloud in temporary_storage (Buffer)* **do**
  - **if** ***Condition 6*** *is satisfied* **then**
    - Remove data-cloud permanently.
  - **end**
- **end**
- **for** *Every existing data-cloud* **do**
  - **if** ***Condition 7*** *is satisfied* **then**
    - Merge data-cloud i^{th} and j^{th} one into i^{th} data cloud.
    - Update properties of i^{th} data-cloud with formulae 27
  - **end**
- **end**
- Create AnYa type data-cloud based fuzzy rule using 18.

end

4.3 Decision Making Procedure

This subsection will describe the procedure for validating a data samples label by using pre-selected data-clouds. In both offline and online training architecture, the constructing procedure of AnYa type data-cloud based fuzzy rule has described.

For the arriving data sample (x), each active fuzzy rule will provide a firing strength by computing the similarity between x and pre-selected data-cloud denoted by $\gamma^c(x)$ (c = 1, 2, 3...C), which will be determined by the following formulae:

$$\gamma^c(x) = \max_{Cloud\_centre \in \{Cloud\_centre^C\}} \left(e^{-d^2(x, Cloud\_centre)} \right) \tag{28}$$

Further, depending on the firing strength of per class (one rule from a class) rules, the class label of x will be set by a straightforward way with general decision-maker using the "winner takes all" formulae as well[7]:

$$\text{class label} = \arg \max_{c=1,2....C} (\gamma^c(x)) \tag{29}$$

4.4 Dataset

To describe calculating efficiency of the newly proposed data-cloud based classifier, well known benchmark dataset will be used and further it will be compared with well-established classifiers namely Evolving classifiers (eclass)[7]), Support vector machine (SVM) classifier[23], K-nearest neighbour (KNN) classifier[24], Decision tree (DT) classifier [46], Self-organising map (SOM) classifier [42], DENFIS classifier [33], SOF[26] classifier and TEDAClass classifier[32].

- **Occupancy detection**: A real-world multi-variate time series dataset on room occupancy (binary) as per the environmental condition of the room[20]. This dataset contains 20560 instances and 7 attributes for two class- occupied(1) and non-occupied(0).
- **Optical recognition**: It contains 5620 instance of 10 class. The 64 feature of Optical recognition dataset is extracted from handwritten digits into bitmaps.
- **Multiple feature**: This dataset consists of features of handwritten numerals (0-9) extracted from a collection of Dutch utility maps. 200 patterns per class (for a total of 2,000 patterns) have been digitized in binary images. For every instance it has 649 feature.
- **Letter Recognition**: This english capital letter recognition dataset contains 20000 instance with 16 features and 26 class labels.

References

[1] Plamen Angelov. "Anomaly detection based on eccentricity analysis". In: *2014 IEEE symposium on evolving and autonomous learning systems (EALS)*. IEEE. 2014, pp. 1–8.

[2] Plamen Angelov. "Typicality distribution function—A new density-based data analytics tool". In: *2015 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2015, pp. 1–8.

[3] Plamen P Angelov. *Evolving rule-based models: a tool for design of flexible adaptive systems*. Vol. 92. Physica, 2013.

[4] Plamen P Angelov and Xiaowei Gu. "Empirical fuzzy sets". In: *International Journal of Intelligent Systems* 33.2 (2018), pp. 362–395.

[5] Plamen P Angelov, Xiaowei Gu, and José C Prıncipe. "A generalized methodology for data analysis". In: *IEEE transactions on cybernetics* 48.10 (2017), pp. 2981–2993.

[6] Plamen P Angelov, Xiaowei Gu, and José C Prıncipe. "Autonomous learning multimodel systems from data streams". In: *IEEE Transactions on Fuzzy Systems* 26.4 (2017), pp. 2213–2224.

[7] Plamen P Angelov and Xiaowei Zhou. "Evolving fuzzy-rule-based classifiers from data streams". In: *Ieee transactions on fuzzy systems* 16.6 (2008), pp. 1462–1475.

[8] Plamen Angelov and Xiaowei Gu. "MICE: Multi-layer multi-model images classifier ensemble". In: *2017 3rd IEEE International Conference on Cybernetics (CYBCONF)*. IEEE. 2017, pp. 1–8.

[9] Plamen Angelov, Xiaowei Gu, and Dmitry Kangin. "Empirical data analytics". In: *International Journal of Intelligent Systems* 32.12 (2017), pp. 1261–1284.

[10] Plamen Angelov and Ronald Yager. "A new type of simplified fuzzy rule-based system". In: *International Journal of General Systems* 41.2 (2012), pp. 163–185.

[11] Plamen Angelov and Ronald Yager. "Simplified fuzzy rule-based systems using non-parametric antecedents and relative data density". In: *2011 IEEE Workshop on Evolving and Adaptive Intelligent Systems (EAIS)*. IEEE. 2011, pp. 62–69.

[12] Plamen Angelov and Xiaowei Zhou. "Evolving fuzzy systems from data streams in real-time". In: *2006 International symposium on evolving fuzzy systems*. IEEE. 2006, pp. 29–35.

[13] Plamen Angelov, Xiaowei Zhou, and Frank Klawonn. "Evolving fuzzy rule-based classifiers". In: *2007 IEEE Symposium on Computational Intelligence in Image and Signal Processing*. IEEE. 2007, pp. 220–225.

[14] Plamen Angelov et al. "Empirical data analysis: a new tool for data analytics". In: *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE. 2016, pp. 000052–000059.

[15] Plamen Angelov et al. "Symbol recognition with a new autonomously evolving classifier autoclass". In: *2014 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS)*. IEEE. 2014, pp. 1–7.

[16] Christopher M Bishop et al. *Neural networks for pattern recognition*. Oxford university press, 1995.

[17] B Boots, A Okabe, and K Sugihara. "Spatial tessellations". In: *Geographical information systems* 1 (1999), pp. 503–526.

[18] Mohamed Anouar Borgi et al. "Regularized shearlet network for face recognition using single sample per person". In: *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2014, pp. 514–518.

[19] Hyeran Byun and Seong-Whan Lee. "Applications of support vector machines for pattern recognition: A survey". In: *International Workshop on Support Vector Machines*. Springer. 2002, pp. 213–236.

[20] Luis M Candanedo and Véronique Feldheim. "Accurate occupancy detection of an office room from light, temperature, humidity and CO2 measurements using statistical learning models". In: *Energy and Buildings* 112 (2016), pp. 28–39.

[21] Peter C Cheeseman et al. "Bayesian Classification." In: *AAAI*. Vol. 88. 1988, pp. 607–611.

[22] Dan Ciregan, Ueli Meier, and Jürgen Schmidhuber. "Multi-column deep neural networks for image classification". In: *2012 IEEE conference on computer vision and pattern recognition*. IEEE. 2012, pp. 3642–3649.

[23] Nello Cristianini, John Shawe-Taylor, et al. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[24] Padraig Cunningham and Sarah Jane Delany. "k-Nearest Neighbour Classifiers–". In: *arXiv preprint arXiv:2004.04523* (2020).

[25] Daniel Gibert et al. "Using convolutional neural networks for classification of malware represented as images". In: *Journal of Computer Virology and Hacking Techniques* 15.1 (2019), pp. 15–28.

[26] Xiaowei Gu and Plamen P Angelov. "Self-organising fuzzy logic classifier". In: *Information Sciences* 447 (2018), pp. 36–51.

[27] Xiaowei Gu et al. "A new type of distance metric and its use for clustering". In: *Evolving Systems* 8.3 (2017), pp. 167–177.

[28] Reihaneh H Hariri, Erik M Fredericks, and Kate M Bowers. "Uncertainty in big data analytics: survey, opportunities, and challenges". In: *Journal of Big Data* 6.1 (2019), p. 44.

[29] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.

[30] Jose Antonio Iglesias et al. "Creating evolving user behavior profiles automatically". In: *IEEE Transactions on Knowledge and Data Engineering* 24.5 (2011), pp. 854–867.

[31] Ruoming Jin and Gagan Agrawal. "Efficient decision tree construction on streaming data". In: *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. 2003, pp. 571–576.

[32] Dmitry Kangin, Plamen Angelov, and José Antonio Iglesias. "Autonomously evolving classifier TEDAClass". In: *Information Sciences* 366 (2016), pp. 1–11.

[33] Nikola K Kasabov and Qun Song. "DENFIS: dynamic evolving neural-fuzzy inference system and its application for time-series prediction". In: *IEEE transactions on Fuzzy Systems* 10.2 (2002), pp. 144–154.

[34] JINHO Kim[1], BS Kim, and Silvio Savarese. "Comparing image classification methods: K-nearest-neighbor and support-vector-machines". In: *Proceedings of the 6th WSEAS international conference on Computer Engineering and Applications, and Proceedings of the 2012 American conference on Applied Mathematics*. Vol. 1001. 2012, pp. 48109–2122.

[35] Tomas Larrain et al. “Face recognition using sparse fingerprint classification algorithm”. In: *IEEE Transactions on Information Forensics and Security* 12.7 (2017), pp. 1646–1657.

[36] Lizhen Lu, Liping Di, and Yanmei Ye. “A decision-tree classifier for extracting transparent plastic-mulched landcover from Landsat-5 TM images”. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 7.11 (2014), pp. 4548–4558.

[37] Ebrahim H Mamdani and Sedrak Assilian. “An experiment in linguistic synthesis with a fuzzy logic controller”. In: *International journal of man-machine studies* 7.1 (1975), pp. 1–13.

[38] Andrew McAfee et al. “Big data: the management revolution”. In: *Harvard business review* 90.10 (2012), pp. 60–68.

[39] Fakhroddin Noorbehbahani et al. “An incremental intrusion detection system using a new semi-supervised stream classification method”. In: *International Journal of Communication Systems* 30.4 (2017), e3002.

[40] Mahesh Pal. “Random forest classifier for remote sensing classification”. In: *International journal of remote sensing* 26.1 (2005), pp. 217–222.

[41] Leif E Peterson. “K-nearest neighbor”. In: *Scholarpedia* 4.2 (2009), p. 1883.

[42] Piotr Płoński and Krzysztof Zaremba. “Self-organising maps for classification with metropolis-hastings algorithm for supervision”. In: *International Conference on Neural Information Processing*. Springer. 2012, pp. 149–156.

[43] Mahardhika Pratama et al. “PANFIS: A novel incremental learning machine”. In: *IEEE Transactions on Neural Networks and Learning Systems* 25.1 (2013), pp. 55–68.

[44] Shahriar Rahman Fahim et al. “Microgrid fault detection and classification: Machine learning based approach, comparison, and reviews”. In: *Energies* 13.13 (2020), p. 3460.

[45] Hai-Jun Rong et al. “Sequential adaptive fuzzy inference system (SAFIS) for nonlinear system identification and prediction”. In: *Fuzzy sets and systems* 157.9 (2006), pp. 1260–1275.

[46] S Rasoul Safavian and David Landgrebe. “A survey of decision tree classifier methodology”. In: *IEEE transactions on systems, man, and cybernetics* 21.3 (1991), pp. 660–674.

[47] Na Su et al. “An arbitrary shape clustering algorithm over variable density data streams”. In: *Journal of Algorithms & Computational Technology* 11.1 (2017), pp. 93–99.

[48] Asit Subudhi, Manasa Dash, and Sukanta Sabut. “Automated segmentation and classification of brain stroke using expectation-maximization and random forest classifier”. In: *Biocybernetics and Biomedical Engineering* 40.1 (2020), pp. 277–289.

[49] Johan AK Suykens and Joos Vandewalle. “Least squares support vector machine classifiers”. In: *Neural processing letters* 9.3 (1999), pp. 293–300.

[50] Tomohiro Takagi and Michio Sugeno. “Fuzzy identification of systems and its applications to modeling and control”. In: *IEEE transactions on systems, man, and cybernetics* 1 (1985), pp. 116–132.

[51] Mingjing Wang and Huiling Chen. “Chaotic multi-swarm whale optimizer boosted support vector machine for medical diagnosis”. In: *Applied Soft Computing* 88 (2020), p. 105946.

[52] Qian Weng et al. “Land-use classification via extreme learning classifier based on deep convolutional features”. In: *IEEE Geoscience and Remote Sensing Letters* 14.5 (2017), pp. 704–708.

[53] Gui-Song Xia et al. “AID: A benchmark data set for performance evaluation of aerial scene classification”. In: *IEEE Transactions on Geoscience and Remote Sensing* 55.7 (2017), pp. 3965–3981.

[54] Wenbin Zhang and Eirini Ntoutsi. “Faht: an adaptive fairness-aware decision tree classifier”. In: *arXiv preprint arXiv:1907.07237* (2019).

Meeting with supervisor

Date	Discussion Topic	Place \Technology	Comment
5 Dec, 2019	Basic Discussion	Dept. of CSE	-
12 Dec, 2019	Data science and research procedure	Dept. of CSE	-
19 Jan, 2020	Data classification algorithms and Research paper selection	Dept. of CSE	-
23 Feb, 2020	Fuzzy rule based and Neuro fuzzy classifier	Dept. of CSE	-
4 Mar, 2020	Fuzzy rule based classifier	Dept. of CSE	-
12 Jun, 2020	Fuzzy rule based classifier structure	Zoom app	-
15 Aug, 2020	Proposed FRB classifier structure	Zoom app	-
28 Sep, 2020	Proposed FRB classifier structure	Zoom app	-
13 Oct, 2020	Proposed FRB classifier structure	Dept. of CSE	-
14 Dec, 2020	Proposed FRB classifier structure	Zoom app	-