Prayatul Matrix for Evaluating Clustering Algorithms: A Direct Comparison Approach

Anupam Biswas, *Member, IEEE*

Abstract—Performance comparison of clustering algorithms is often done in terms of different confusion matrix based scores obtained on test datasets when ground truth is available. However, a dataset comprises several instances having different difficulty levels. Therefore, it is more logical to compare effectiveness of clustering algorithms on individual instances instead of comparing scores obtained for the entire dataset. In this paper, an alternative approach is proposed for direct comparison of clustering algorithms in terms of individual instances within the dataset. A direct comparison matrix called Prayatul Matrix is prepared, which accounts for comparative outcome of two clustering algorithms on different instances of a dataset. Five different performance measures are designed based on prayatul matrix. Theoretical analysis shows proposed measures satisfy five important properties such as scale invariance, data invariance, permutation invariance, monotonicity and continuity. The efficacy of proposed approach as well as designed measures is analyzed empirically with four clustering algorithms on widely used standard datasets. Indications of the proposed measures are compared with confusion matrix-based measures as well as other three permutation invariant measures. Results are evident that the newly designed measures are capable of giving some important insight about the clustering algorithms, which were impossible with existing measures.

Index Terms—Machine Learning, Unsupervised Learning, Clustering, Direct Comparison, Performance Measures.

◆

1 INTRODUCTION

CONFUSION matrix has been the central element of performance evaluation of Machine Learning (ML) models, irrespective of application domains [1], [2], [3], [4], [5]. A confusion matrix is a kind of contingency table, where each row represents an actual class, while each column represents a predicted class. Be it classification [6] or clustering algorithm [7], confusion matrix is prepared to evaluate and visually describe the performance of the model on a test dataset for which the ground truth labels are known. For evaluating clustering algorithms in particular, different measures are computed based on confusion matrix to enumerate the performance. The confusion matrix or the measures that are computed based on confusion matrix are single model driven. Thus, to compare performance of two clustering algorithms, two separate confusion matrices have to be prepared for each algorithm and relevant measures have to be computed. One of the major drawbacks of this kind of performance comparison is it lacks direct comparison of algorithms on individual instances of the dataset. For instance, one can compare true positive value of one confusion matrix with another, which determines how many times two algorithms are correct. However, it cannot determine exactly on which instances the algorithms are correct or whether the algorithms are correct on the same or different instances. Important to note that both the algorithms can have the same true positive values but instance-wise those may be completely opposite. Therefore, the same true positive values or accuracy or any other measures designed based on confusion matrix do not mean that performance of both the algorithms will be same.

Though, different measures are available that are not based on confusion matrix [1], [8], mostly share the same drawback as mentioned above. Specifically for clustering, several intrinsic measures [9], [10] are designed that determines the quality of clustering. However, these measures also exhibit the same issue as they neither indicate direct comparison at instance level nor they consider instance level outcome for comparison. In this paper, an alternative approach is proposed that enables direct comparison of clustering algorithms at the level of instances within the dataset. A direct comparison matrix called *Prayatul Matrix* is prepared for accounting as well as visualizing the comparative outcome of two clustering algorithms. The key features of the proposed approach are as follows:

- Pairwise instance level comparative outcome of two clustering algorithms is presented in a direct comparison matrix called *prayatul matrix*.
- Instance level outcomes of both the clustering algorithms are compared pairwise in reference to the ground truth.
- Five performance measures are designed based on the elements of prayatul matrix, which indicate a direct comparative scores between two clustering algorithms.
- All five measures satisfy important properties such as scale invariance, data invariance, permutation invariance, monotonicity and continuity.

Rest of the paper is organized as follows. Section 2 discusses the works related to contingency matrix based measures. Section 3 elaborates the proposed direct comparison approach for evaluating clustering algorithms, the

---

- *A. Biswas was with the Department of Computer Science and Engineering, National Institute of Technology Silchar, Assam-788010, India. E-mail: anupam@cse.nits.ac.in*

Manuscript received Month xx, 20yy; revised Month xx, 20yy.

prayatul matrix and measures designed. Section 4 details about experimental analysis covering experimental setup, datasets, and result analysis. Section 5 concludes highlighting the key advantages of the proposed approach.

2 RELATED WORK

The origin of widely used confusion matrix for ML models can be traced back to Kerl Pearson's work [11], where it was referred as contingency table. Later on the term confusion matrix [12] was used in the context of psychology. While the confusion matrix used in ML to evaluate both classification [6] and clustering algorithms [7] is prepared on the basis of ground truth labels and predicted labels. The conventional way to compare accuracy of two clustering algorithms in particular is to prepare two separate confusion matrices for each model and generate scores to compare. One of the major drawbacks of this approach is it lacks direct comparison of clustering algorithms on individual instances of the dataset. Moreover, clustering labels obtained with clustering algorithms are arbitrary. Thus, often pair confusion matrix [13] is prepared instead of confusion matrix. Different measures like Adjusted Rand Index (ARI) [14], Normalized Mutual Information (NMI) [15] and Adjusted Mutual Information (AMI) [16] are designed to deal with arbitrary cluster labels. However, these attempts also lack direct comparison at instance level.

The proposed direct comparison approach prepares a single comparison matrix by comparing pair of instance level outcomes of two clustering algorithms w.r.t. ground truth. Earlier Dietterich [17] demonstrated a similar idea to construct comparison matrix for misclassified instances two ML models, particularly for the classification algorithms as an application of McNemar's test [18]. However, Dietterich's approach focuses simply on misclassified instances limited to supervised learning only, and analysis is done under the null hypothesis *the two algorithms should have the same error rate*. Whereas, proposed approach considers all instances to prepare the comparison matrix and analysis is done on the basis of scores not null hypothesis.

In recent years, few attempts are being made to prepare confusion matrix alternatively. A three-way confusion matrix is designed that visualizes the degree of algorithm confusion within different classes [19]. The construction of basic probability assignment (BPA) based on the confusion matrix has also been studied in the context of classification problem [20]. Simplified confusion matrix visualization techniques are also designed for better visualization of classes [21], [22]. However, none of these approaches considered direct comparison of ML models at instance level and are specifically designed for classification algorithms.

3 DIRECT COMPARISON APPROACH

The direct comparison approach involves two clustering algorithms in the process. The role of the participating clustering algorithms in the direct comparison approach are defined as follows:

Definition 1 (Primary Algorithm (A_p))**.** The algorithm whose performance is to be evaluated in comparison to other algorithm is referred as primary algorithm.

		Alternative		
		Right	**Wrong**	**Total**
Primary	**Right**	Both Right	Right Wrong	BR+RW
	Wrong	Wrong Right	Both Wrong	WR+BW
Total		BR+WR	RW+BW	$\frac{1}{2}N(N-1)$

Fig. 1: Prayatul Matrix for two clustering algorithms (Primary and Alternative) with different abstractions

Definition 2 (Alternative Algorithm (A_q))**.** The algorithms with whom the primary algorithm is to be compared is referred as alternative algorithm.

The primary algorithm can be compared with multiple alternatives on same or different datasets. Let us consider a test dataset having N instances with ground truth labels $G = \{g_1, g_2, g_3, ..., g_N\}$. Let us consider N predicted cluster labels obtained for N different instances of test dataset with primary algorithm A_p and alternative algorithm A_q are $P = \{p_1, p_2, p_3, ...p_N\}$ and $Q = \{q_1, q_2, q_3, ...q_N\}$ respectively.

Since clustering algorithms assign the labels arbitrarily to the instances of test dataset, instead of comparing the labels of instances, sameness of labels for the pair of instances are compared. The sameness of predicted labels of all pairs of instances of A_p are compared directly with that of A_q w.r.t. ground truth. Thus, $N(N-1)/2$ number of pairs obtained for N different instances of test dataset considered for direct comparison of instance pairs in terms of sameness of the predicted labels.

3.1 Prayatul Matrix

A 2×2 dimensional direct comparison matrix $\mathcal{D}$ named *Prayatul Matrix* is prepared by comparing the instances pairwise in terms of sameness of the predicted labels of A_p with that of A_q w.r.t. ground truth G of the test dataset. The prayatul matrix is a kind of contingency table that has two levels of abstractions *Right* and *Wrong* both in rows and columns, which indicate the correctness of outcomes obtained with both primary and alternative algorithms pairwise. The abstractions *Right* and *Wrong* means a pair of outcomes of the algorithm is correct and incorrect respectively w.r.t. G. Abstractions related to primary and alternative algorithms are placed in rows and columns respectively as shown in Fig. 1.

Given the arbitrary predicted cluster labels, if the instance pair belongs to same cluster and the ground truth labels of that pair of instances also imply both belong to same cluster then the predicted labels of instance pair is considered as a match with the ground truth i.e. abstraction

Right. Similarly, if the instance pair belongs to different clusters and the ground truth labels of that pair of instances also imply both belong to different cluster then the predicted labels of instance pair is considered as a match with the ground truth i.e. abstraction *Right*. Otherwise, the predicted labels of instance pair is considered as mismatch i.e. abstraction *Wrong*. Let R_p and R_q respectively are the set of instance pairs where predicted labels of A_p and A_q are correct i.e. the predicted labels of instance pairs match the ground truth so the pair comes under the abstraction *Right*. Let W_p and W_q respectively are the set of instance pairs where prediction of A_p and A_q are incorrect i.e. the predicted labels of instance pairs mismatch the ground truth so the pair comes under the abstraction *Wrong*. Now, the instance pairs for which both A_p and A_q are *Right* is given by $\{R_p \cap R_q\}$ and the corresponding entry for the matrix $\mathcal{D}$ is computed as follows:

$$\mathcal{D}_{11} = |R_p \cap R_q| \tag{1}$$

Likewise, the instance pairs for which A_p is *Right* but A_q is *Wrong* is given by $\{R_p \cap W_q\}$ and the corresponding entry for the matrix $\mathcal{D}$ is computed as follows:

$$\mathcal{D}_{12} = |R_p \cap W_q| \tag{2}$$

The instance pairs for which A_p is *Wrong* but A_q is *Right* is given by $\{W_p \cap R_q\}$ and the corresponding entry for the matrix $\mathcal{D}$ is computed as follows:

$$\mathcal{D}_{21} = |W_p \cap R_q| \tag{3}$$

Lastly, the instance pairs for which both A_p and A_q are *Wrong* is given by $\{W_p \cap W_q\}$ and the corresponding entry for the matrix $\mathcal{D}$ is computed as follows:

$$\mathcal{D}_{22} = |W_p \cap W_q| \tag{4}$$

Interpretation of different elements of the prayatul matrix $\mathcal{D}$ for the instance pairs of test dataset in terms of abstractions *Right* and *Wrong* are done as follows:

- **Both Right (BR):** The instance pairs for which both A_p and A_q are right w.r.t. G.
- **Right Wrong (RW):** The instance pairs for which A_p is right but A_q is wrong w.r.t. G.
- **Wrong Right (WR):** The instance pairs for which A_p is wrong but A_q is right w.r.t. G.
- **Both Wrong (BW):** The instance pairs for which both A_p and A_q are wrong w.r.t. G.

3.2 Comparative Performance Measures

The elements of the prayatul matrix i.e. BR, RW, WR and BW are used to design five comparative performance measures for a pair of clustering algorithms A_p and A_q as follows:

The elements RW and WR are the counts of instance pairs, where both algorithms are having disagreement i.e. deviates from each others decision. If RW count is more that means primary algorithms is better in taking right decisions compared to alternative and it means opposite if WR is more. Subtractions of WR from RW penalizes the wrong decisions of primary algorithm. Normalizing it with all deviating instance pair counts i.e. RW+ WR gives the comparative deviation. This measure indicates how two algorithms are deviating from each other when outcomes of both are different. Positive value implies primary algorithm is better, while negative value implies alternative algorithm is better in terms of right outcomes. Formally, the comparative deviation of A_p and A_q is defined as follows:

Definition 3 (Comparative Deviation (σ_c))**.** The comparative deviation of primary algorithm over alternative algorithm is defined as:

$$\sigma_c(P, Q) = \frac{RW - WR}{RW + WR} \tag{5}$$

On the other hand, the elements BR and BW are the counts of instance pairs, where both algorithms agree. If BR is more then it means both algorithms are polarized towards right decision, whereas it means opposite if BW is more. Subtractions of BR from BW penalizes the wrong decisions of both algorithms. Addition of RW with BR-BW gives the polarization of primary algorithm towards right decision in comparison to alternative algorithm. Normalizing it with the total paired outcome counts gives the polarization of primary algorithm. This measure indicates how the primary algorithm is polarized towards right or wrong decision. Positive value implies primary algorithm is good at taking right decision and negative implies bad at taking right decision in comparison to alternative. The polarization of A_p in comparison to A_q is defined as follows:

Definition 4 (Polarization (α))**.** The polarization of primary algorithm and alternative algorithm is defined as:

$$\alpha(P, Q) = \frac{BR + RW - BW}{BR + RW + WR + BW} \tag{6}$$

The elements BR and RW together gives the count of instance pairs where primary algorithm is right. Normalizing it with the count of instance pairs where at least one of the algorithms is right (i.e. BR + RW + WR) gives the comparative rightness of primary algorithm. While penalizing wrong decisions of primary gives the effective rightness of primary algorithm. Formally, comparative rightness and effective rightness of A_p in comparison to A_q are defined as follows:

Definition 5 (Comparative Rightness (ξ_c))**.** The comparative rightness of primary algorithm over alternative algorithm is defined as:

$$\xi_c(P, Q) = \frac{BR + RW}{BR + RW + WR} \tag{7}$$

Definition 6 (Effective Rightness (ξ_e))**.** The effective rightness of primary algorithm over alternative algorithm is defined as:

$$\xi_e(P, Q) = \frac{BR + RW - WR}{BR + RW + WR} \tag{8}$$

Higher ξ_c and ξ_e values indicate primary algorithm is good at taking right decisions and primary algorithm is good at taking right decisions despite of its wrong decisions respectively.

Effective rightness measure indicates how good the primary algorithm is on taking right decision considering all right decisions and penalizing its wrong decision. However, it is from the perspective of all decisions where at least one of the algorithms is right. To have superiority over alternative algorithm, the primary algorithm has to perform better from the perspective of all decisions. Thus, effective superiority of primary algorithm in comparison to alternative is defined as follows:

Definition 7 (Effective Superiority (ϕ_e))**.** The effective superiority of primary algorithm over alternative algorithm is defined as:

$$\phi_e(P, Q) = \frac{BR + RW - WR}{BR + RW + WR + BW} \tag{9}$$

Higher ϕ_e value indicates primary algorithm is superior at taking right decisions in comparison to alternative algorithm.

Theorem 1. *The measures σ_c, α, ξ_e and ϕ_e have the range [-1, +1], but ξ_c has the range [0, 1].*

Proof. The proof is quite straight forward. Since all the measures except ξ_c has a negative element in numerator, which can have maximum $N(N-1)/2$ value and apparently other elements will be 0 in that case, implying minimum value -1. While both the elements in numerator of ξ_c can have 0 simultaneously implying minimum value 0. For all measures, one positive element in numerator can have maximum $N(N-1)/2$ value and apparently other elements will be 0 in that case, implying maximum value +1. ☐

3.3 Properties of Comparative Performance Measures

Unlike the measures that are defined based on confusion matrix indicate the performance of a standalone algorithm, the five measures defined based on prayatul matrix indicate comparative performance of one algorithm over another. The properties of proposed comparative performance measures are analyzed theoretically in the context of following five axioms:

- **Scale Invariance:** Metric should scale irrespective of sample size small or large.
- **Data Invariance:** Metric should not be affected by unbalances within the dataset.
- **Permutation Invariance:** Metric should not be affected by permutation of labels.
- **Monotonicity:** Metric has to be non-decreasing under monotonic consistent improvement.
- **Continuity:** Small change in #samples N has to cause a smaller impact on the metric.

Theorem 2. *All five measures $\sigma_c, \alpha, \xi_c, \xi_e$ and ϕ_e are scale invariant.*

Proof. All the proposed measures are multivariate functions only. Let us consider, the case of σ_c. By replacing RW and WR with variables x_1 and x_2 respectively, σ_c can be written in the form of a function as follows:

$$\sigma_c(x_1, x_2) = \frac{x_1 - x_2}{x_1 + x_2}. \tag{10}$$

A multivariate function $f(x_1, x_2, ...)$ said to be scale invariant if it satisfies

$$f(\lambda_1 x_1, \lambda_2 x_2, ...) = C(\lambda_1, \lambda_2, ...) f(x_1, x_2, ...). \tag{11}$$

Since, RW and WR are dependent on the predictions of primary and alternatives i.e. P and Q, change in size of the test samples will imply change in RW and WR. Let the changes in RW and WR be obtained as factors of λ_1 and λ_2. Thus, $\sigma_c(x_1, x_2)$ will be scale-invariant if power-low dependency can be shown considering the following

$$\sigma_c(\lambda_1 x_1, \lambda_2 x_2) = \frac{\lambda_1 x_1 - \lambda_2 x_2}{\lambda_1 x_1 + \lambda_2 x_2} \tag{12}$$

where, λ_1 and λ_2 are the factors for RW and WR resulted in due to change in size of the test samples. First, taking the logarithm of both sides yields

$$\ln \sigma_c(\lambda_1 x_1, \lambda_2 x_2) = \ln\left(\frac{\lambda_1 x_1 - \lambda_2 x_2}{\lambda_1 x_1 + \lambda_2 x_2}\right). \tag{13}$$

Introducing a new function $F(x)$ defined as $F(\ln x) = \sigma_c(x)$ to above equation gives

$$\begin{aligned}
&\ln F(\ln \lambda_1 + \ln x_1, \ln \lambda_2 + \ln x_2) = \ln\left(\frac{\lambda_1 x_1 - \lambda_2 x_2}{\lambda_1 x_1 + \lambda_2 x_2}\right) \\
&= \ln(\lambda_1 x_1 - \lambda_2 x_2) - \ln(\lambda_1 x_1 + \lambda_2 x_2) \\
&= \ln \lambda_1 + \ln x_1 - \ln \lambda_2 - \ln x_2 - \ln \lambda_1 - \ln x_1 - \ln \lambda_2 - \ln x_2 \\
&= \ln \lambda_1 - \ln \lambda_2 - \ln \lambda_1 - \ln \lambda_2 + \ln x_1 - \ln x_2 - \ln x_1 - \ln x_2 \\
&= \ln(\lambda_1 - \lambda_2) - \ln(\lambda_1 + \lambda_2) + \ln(x_1 - x_2) - \ln(x_1 + x_2) \\
&= \ln\left(\frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2}\right) + \ln\left(\frac{x_1 - x_2}{x_1 + x_2}\right) \\
&= \ln\left(\frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2}\right)\left(\frac{x_1 - x_2}{x_1 + x_2}\right).
\end{aligned}$$

Finally, the equation becomes

$$\ln F(\ln \lambda_1 + \ln x_1, \ln \lambda_2 + \ln x_2) = \ln\left(\frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2}\right)\left(\frac{x_1 - x_2}{x_1 + x_2}\right). \tag{14}$$

Now, taking inverse function both sides yield

$$e^{F(\ln \lambda_1 + \ln x_1, \ln \lambda_2 + \ln x_2)} = e^{\left(\frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2}\right)\left(\frac{x_1 - x_2}{x_1 + x_2}\right)}. \tag{15}$$

Applying logarithm on both sides to above equation gives

$$F(\ln \lambda_1 + \ln x_1, \ln \lambda_2 + \ln x_2) = \left(\frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2}\right)\left(\frac{x_1 - x_2}{x_1 + x_2}\right). \tag{16}$$

Changing the function back to $\sigma_c(x)$ gives

$$\sigma_c(\lambda_1 x_1, \lambda_2 x_2) = \left(\frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2}\right)\left(\frac{x_1 - x_2}{x_1 + x_2}\right). \tag{17}$$

Representing right side as functions yields

$$\sigma_c(\lambda_1 x_1, \lambda_2 x_2) = C(\lambda_1, \lambda_2)\sigma_c(x_1, x_2) \tag{18}$$

where, C is a function of two variables λ_1 and λ_2. Hence, proved that σ_c is scale invariant. Similarly, measures α, ξ_c, ξ_e and ϕ_e can also be proven as scale invariant. ☐

Theorem 3. *All five measures $\sigma_c, \alpha, \xi_c, \xi_e$ and ϕ_e are data invariant.*

Proof. Since the measures do not directly depend on the number of samples in each class or sequence in which samples are considered, balance or unbalance dataset, it does not have any direct impact on the measures. Moreover, measure considers the elements of prayatul matrix, which are counts of sameness of instance pairs for comparative outcomes of two algorithms. Therefore, even if the dataset is unbalanced, it will not have any impact on the measures. For instance, the measure σ_c has two elements RW and WR. The element RW will be influenced only when the outcomes of primary algorithm is right and alternative algorithm is wrong for the same instances of the dataset it does not matter which instance pairs belong or even all the instance pairs may belong to single cluster. Likewise, element WR will be influenced only when the outcomes of primary algorithm is wrong and alternative algorithm is right for the same instances of the dataset. Same is the case for the elements BR and BW. Therefore, all five measures σ_c, α, ξ_c, ξ_e and ϕ_e are data invariant. ☐

Theorem 4. All five measures $\sigma_c, \alpha, \xi_c, \xi_e$ and ϕ_e are permutation invariant.

Proof. Let us consider instances i and j have the same ground truth labels g_l i.e. $g_i = g_j = g_l$. Say, predicted labels for instances i and j with primary and alternative algorithms are also same i.e. $p_i = p_j = p_l$ and $q_i = q_j = q_l$. Thus, the instance pair i and j will lead to increment of BR by one in the prayatul matrix, since predicted labels for instances i and j with primary and alternative algorithms are same and ground truth labels are also same. Important to note that, predicted labels p_l and q_l of instance i or j are not compared the ground truth label g_l. Just checked the sameness of the labels of instance pairs.

As the instances are expected to have arbitrary clustering labels, so the instance pair i and j can also have any arbitrary labels p_l' and q_l' instead of p_l and q_l respectively for primary and alternative algorithms. However, as long as $p_i = p_j$ and $q_i = q_j$ remains true for instance pair i and j, only BR value will change. Similarly, any instance pair affecting other elements RW, WR or BW remain unaltered by any arbitrary permutation of labels as long as sameness or difference of labels for the instance pair is preserved.

Since, all five measures $\sigma_c, \alpha, \xi_c, \xi_e$ and ϕ_e are designed based on the elements of prayatul matrix only, so all are permutation invariant. ☐

Theorem 5. All five measures $\sigma_c, \alpha, \xi_c, \xi_e$ and ϕ_e are monotonic.

Proof. Consistent improvement of σ_c, α, ξ_c, ξ_e and ϕ_e in the context primary algorithm means non-decreasing changes in these measures. Consistent improvement can happen either when the primary algorithm is right or alternative algorithm is wrong in majority of the instance pairs i.e. the prayatul matrix elements which involves abstraction $Right$ for primary is more or abstraction $Wrong$ for alternative is more. Likewise, consistent improvement can also happen when the prayatul matrix elements which involves abstraction $Wrong$ for primary is less or abstraction $Right$ for alternative is less.

Now, considering the measure σ_c, where numerator is sum of a positive RW and negative WR. Thus, increment of RW or decrement of WR implies consistent improvement of σ_c and it will have non-decreasing values. Similarly, increment of BR and RW or decrement of BW implies consistent improvement of α and the values will be non-decreasing as well. In the same way, consistent improvement of ξ_c, ξ_e and ϕ_e will happen if BR and/or RW increases or if WR and/or BW decreases. Therefore, σ_c, α, ξ_c, ξ_e and ϕ_e, all are non-decreasing under monotonic consistent improvement. ☐

Theorem 6. All five measures $\sigma_c, \alpha, \xi_c, \xi_e$ and ϕ_e are continuous.

Proof. For continuity of a measure, small changes in number of sample has to cause a smaller impact on it. Let us consider a new instance is added in the test dataset. Now, let us examine the impact of newly added instance on each of the elements of the prayatul matrix and on all five measures. Clearly, by the definition of prayatul matrix, only one of the values among BR, RW, WR and BW will be increased or decreased by 1 for every pair of instances, while other three will remain same. Since, the newly added instance will be paired with N existing instances, there will be N such changes.

Considering, the measure σ_c which involves only two elements of the prayatul matrix i.e. RW and WR. Thus, there has the possibility that both RW and WR may remain unchanged for any instance pair. If any one of RW and WR increase by 1, we will have

$$\sigma_c(P,Q) = \begin{cases} \frac{RW-WR+1}{RW+WR+1} & \text{if RW increases} \\ \\ \frac{RW-WR-1}{RW+WR+1} & \text{if WR increases} \end{cases} \tag{19}$$

Replacing RW-WR and RW+WR by a and b the above becomes

$$\sigma_c(P,Q) = \begin{cases} \frac{a-1}{b+1} & \text{if RW increases} \\ \\ \frac{a-1}{b+1} & \text{if WR increases} \end{cases} \tag{20}$$

By Theorem 1 we have $a \leq b$ always, which implies $a+1 \leq b+1$ as well as $a-1 < b+1$. If $a >> 1$ and $b >> 1$ then by properties of ratio,

$$\begin{cases} \dfrac{a+1}{b+1} \equiv \dfrac{a}{b} \\ \dfrac{a-1}{b+1} \equiv \dfrac{a}{b} \end{cases} \tag{21}$$

Same is true for additional $N-1$ number of instances if $a >> N$ and $b >> N$. Since, both a and b can have maximum $N(N-1)/2$, which grows exponentially with N, so $a >> N$ and $b >> N$ will be true from large N. Thus, small change in number of samples σ_c will have minimal so it is continuous. Similarly, the other measures α, ξ_c, ξ_e and ϕ_e can also be proven as continuous. ☐

3.4 Prayatul Matrix Generation

Prayatul matrix generation process is quite simple and straight forward. A simple algorithm called *$\mathcal{D}$-Matrix Algorithm* is designed for generating prayatul matrix as shown in the Algorithm 1. The algorithm takes three inputs: ground truth G, outcome P of the primary algorithm and outcome Q of the alternative algorithm. The entries of prayatul matrix is computed based on the abstraction levels as specified above and finally the algorithm returns the prayatul matrix

D. The time complexity of the algorithm is $\mathcal{O}(N^2)$, where N is the number instances.

Algorithm 1: $\mathcal{D}$-Matrix Algorithm

```
Input: $G, P, Q$
Output: Prayatul Matrix ($\mathcal{D}$)
procedure generatePrayatulMatrixCluster($G, P, Q$)
    $\mathcal{D}_{ij} \leftarrow 0, \forall i, j \in [1, 2]$
    $N \leftarrow$number of instances in $G$
    for $i = 1$ to $N$ do
        for $j = i + 1$ to $N$ do
            if $(p_i = p_j$ AND $q_i = q_j$ AND $g_i = g_j)$ OR $(p_i \neq p_j$ AND $q_i \neq q_j$ AND $g_i \neq g_j)$ then $\mathcal{D}_{11} \leftarrow \mathcal{D}_{11} + 1$
            else if $(p_i = p_j$ AND $q_i \neq q_j$ AND $g_i = g_j)$ OR $(p_i \neq p_j$ AND $q_i = q_j$ AND $g_i \neq g_j)$ then $\mathcal{D}_{12} \leftarrow \mathcal{D}_{12} + 1$
            else if $(p_i \neq p_j$ AND $q_i = q_j$ AND $g_i = g_j)$ OR $(p_i = p_j$ AND $q_i \neq q_j$ AND $g_i \neq g_j)$ then $\mathcal{D}_{21} \leftarrow \mathcal{D}_{21} + 1$
            else if $(p_i = p_j$ AND $q_i = q_j$ AND $g_i \neq g_j)$ OR $(p_i \neq p_j$ AND $q_i \neq q_j$ AND $g_i = g_j)$ then $\mathcal{D}_{22} \leftarrow \mathcal{D}_{22} + 1$
    return $\mathcal{D}$
```

4 EXPERIMENTAL ANALYSIS

4.1 Experimental Setup

4.1.1 Datasets

To analyze efficacy of the proposed prayatul matrix based method as well as newly designed performance measures, three widely used standard datasets Noisy Circles (NC), Noisy Moons (NM), and Aniso (NS) are considered from Scikit-learn package [23]. All of the three datasets contain 1500 instances. Noisy Circles dataset is generated using Make Circles base dataset with noise=0.05 and cluster parameter values as damping=0.77, preference=-240, quantile=0.2, n_clusters=2, min_samples=20, and ξ=0.25. Similarly, Noisy Circles dataset is generated using Make Circles base dataset with noise=0.05 and cluster parameter values as ϵ=0.15, n_neighbors=2, min_samples=20, ξ=0.1, and min_cluster_size=0.2. Likewise, Aniso dataset is generated using Make Bloobs base dataset with transformation = [[0.6, -0.6], [-0.4, 0.8]] and cluster parameter values as ϵ=0.15, n_neighbors=2, min_samples=20, ξ=0.1, and min_cluster_size=0.2.

4.1.2 Clustering Algorithms

Four popular clustering algorithms namely Spectral, K-Means (KMeans), DBSCAN, and BIRCH are considered for unsupervised learning. Specific parameters related to clustering algorithms are set as follows. For KMeans n_clusters=2 or 3, for DBSCAN ϵ=0.3, for Spectral n_clusters=2 or 3, eigen_solver="arpack", affinity="nearest_neighbors", and for BIRCH n_clusters=2 or 3 are considered for the experiments.

4.1.3 Implementation Details and System Configuration

All implementations and executions are done under Jupiter Notebook server 6.4.10 environment with Python 3.10.4. The D-Matrix Algorithm for generating prayatul matrix and proposed performances measures are implemented in Python language [1]. As mentioned above, widely used clustering algorithms that are already implemented and openly available in Scikit-learn package [23] are considered. Publicly available source code for clustering [24] is considered as reference to setup the experimental environment. All the experiments are done on the Computer having Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz with 8 Cores, 4.6GHz Speed, NVIDIA GeForce MX130 Graphics card, 16 GB RAM, 1TB HDD and 64-bit (AMD) Windows 10 Operating System.

4.2 Result Analysis

The proposed direct comparison measures obtained with the four clustering algorithms are analyzed from the perspective measure values as well as instance level comparison through prayatul matrix. Proposed measures are compared with the indications of existing measures and then re-verified with the instance level comparison entries in the prayatul matrix.

4.2.1 Measure Value-based Analysis

It is clear from the results presented in Fig. 2 that DBSCAN and Spectral produced exactly the same clusters as in the ground truth for Noisy Circles and Noisy Moons datasets. However, cluster labels are opposite in Noisy Circles dataset for Spectral, although it produced exactly same clusters as in the ground truth. In the case of BIRCH and KMeans, almost half of the instances are wrongly clustered for Noisy Circles. In Noisy Moons also cluster labels are opposite w.r.t. ground truth for DBSCAN and Spectral, though both produces exactly the same clusters as in ground truth. While all four algorithms misclustered few instances in Aniso dataset, though misclustered instances are comparatively low for DBSCAN and Spectral. With the preliminary visual inspection of results, let us now examine the effectiveness of the measures designed for proposed direct comparison approach.

Since, the proposed direct comparison approach pairs two algorithms for computing performance measure values, the interpretation of these measures are also to be done in the context of pair of two algorithms. Pair of two instances from each algorithm of the two comparing algorithms are considered for direct comparison. The measure values obtained with one-to-one comparison of algorithms are presented in Table 1. For the pair DBSCAN vs Spectral, comparative deviation value is 0 for both Noisy Circles and Noisy Moons, which indicates that there is no difference between the two when there is disagreement on the outcomes. Polarization value is 1 for both Noisy Circles and Noisy Moons for both DBSCAN vs Spectral and Spectral vs DBSCAN, which indicates that both are highly polarized towards right decision. Comparative rightness, effective rightness and effective superiority value is also 1

---

[1] Source codes of prayatul matrix and five scores to be released through GitHub under GPLv3 License

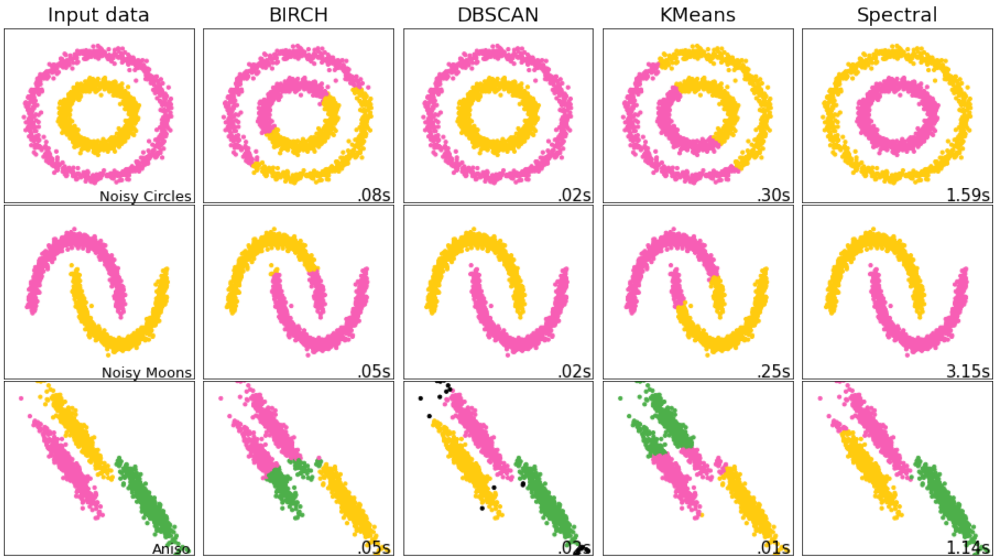


Fig. 2: Results with clustering algorithms

TABLE 1: Proposed measures obtained for one-to-one comparison of each algorithm with three other clustering algorithms

	Algos	DBSCAN					KMeans					Spectral				
BIRCH	Data	σ_c	α	ξ_c	ξ_e	ϕ_c	σ_c	α	ξ_c	ξ_e	ϕ_c	σ_c	α	ξ_c	ξ_e	ϕ_c
	NC	-1	0.5028	0.5028	0.0055	0.0055	0.0062	0.2538	0.6695	0.3389	0.2545	-1	0.5028	0.5028	0.0055	0.0055
	NM	-1	0.8252	0.8252	0.6505	0.6505	0.5565	0.6829	0.9622	0.9244	0.7928	-1	0.8252	0.8252	0.6505	0.6505
	NS	-0.9068	0.7981	0.8005	0.6009	0.6002	-0.1785	0.6962	0.8915	0.7829	0.7021	-0.9396	0.7869	0.8095	0.6190	0.6113
DBSCAN	Algos	BIRCH					KMeans					Spectral				
	Data	σ_c	α	ξ_c	ξ_e	ϕ_c	σ_c	α	ξ_c	ξ_e	ϕ_c	σ_c	α	ξ_c	ξ_e	ϕ_c
	NC	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
	NM	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
	NS	0.9068	0.9877	0.9902	0.9805	0.9792	0.8976	0.9871	0.9909	0.9817	0.9798	0.2639	0.9881	0.9899	0.9797	0.9788
KMeans	Algos	BIRCH					DBSCAN					Spectral				
	Data	σ_c	α	ξ_c	ξ_e	ϕ_c	σ_c	α	ξ_c	ξ_e	ϕ_c	σ_c	α	ξ_c	ξ_e	ϕ_c
	NC	-0.0062	0.2507	0.6653	0.3307	0.2483	-1	0.4997	0.4997	-0.0007	-0.0007	-1	0.4997	0.4997	-0.0007	-0.0007
	NM	-0.5565	0.6016	0.8674	0.7347	0.6301	-1	0.7439	0.7439	0.4878	0.4878	-1	0.7439	0.7439	0.4878	0.4878
	NS	0.1785	0.7257	0.9243	0.8487	0.7611	-0.8976	0.8270	0.8305	0.6609	0.6597	-0.9474	0.8149	0.8407	0.6815	0.6719
Spectral	Algos	BIRCH					KMeans					DBSCAN				
	Data	σ_c	α	ξ_c	ξ_e	ϕ_c	σ_c	α	ξ_c	ξ_e	ϕ_c	σ_c	α	ξ_c	ξ_e	ϕ_c
	NC	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
	NM	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
	NS	0.9396	0.9693	0.9941	0.9881	0.9758	0.9479	0.9676	0.9956	0.9913	0.9774	-0.2639	0.9808	0.9826	0.9652	0.9643

TABLE 2: Accuracy, Precision and Recall based on confusion matrix for clustering algorithms

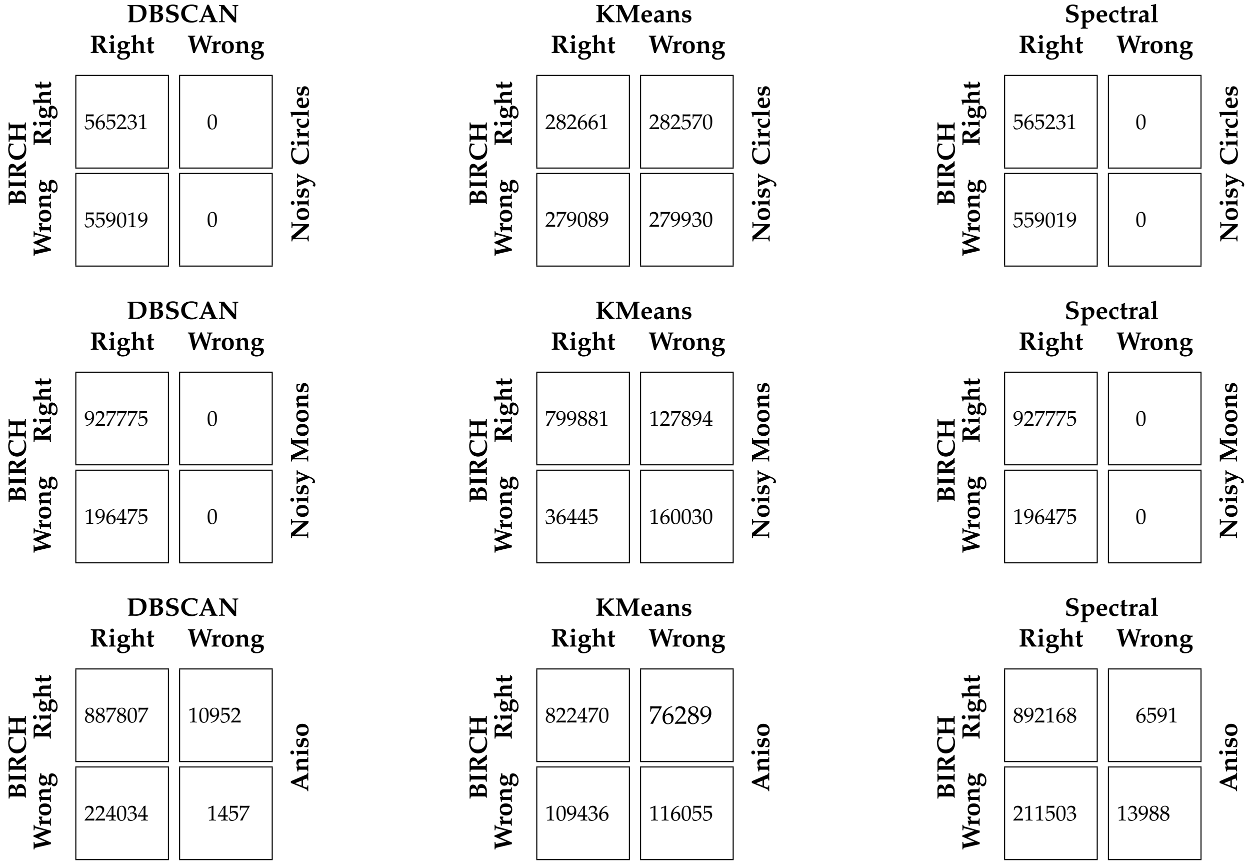
Algos	BIRCH			DBSCAN			KMeans			Spectral		
Data	*Acc*	*Pre*	*Rec*	*Acc*	*Pre*	*Rec*	*Acc*	*Pre*	*Rec*	*Acc*	*Pre*	*Rec*
NC	0.5393	0.5417	0.5107	1	1	1	0.5	0.5	0.5013	0	0	0
NM	0.0967	0.0129	0.0107	0	0	0	0.8493	0.8447	0.8560	0	0	0
NS	0.1947	0.1361	0.1947	0.3253	0.2	0.3253	0.25	0.2520	0.25	0.3467	0.3455	0.3467

for both DBSCAN vs Spectral and Spectral vs DBSCAN, which indicates that both algorithms produce exactly same clusters as noted during visual analysis. Even though cluster labels are opposite w.r.t. ground truth still the measures indicate that Spectral produced exactly the same clusters as in the ground truth. On the other hand, confusion matrix based measures presented in Table 2 failed miserably to capture this fact as Accuracy, Precision and Recall values are 0. Whereas, ARI, NMI and AMI presented in Table 3 clearly indicate that both DBSCAN and Spectral produced exactly the same clusters as in the ground truth.

For BIRCH vs KMeans, comparative deviation value is positive but low in Noisy Circles and Noisy Moons datasets, which indicates BIRCH is capable of taking right decision on some instances where KMeans is wrong. Alternatively, negative values for KMeans vs BIRCH also indicate the same. However, in Aniso dataset, KMeans seems to be better than BIRCH. Polarization value is positive but relatively low for all the datasets, which means both BIRCH and KMeans are less polarized towards right decision. Clearly, relatively low values of effective rightness and effective superiority in Noisy Circles indicate that BIRCH performance is poor specially when its wrong decisions are penalized. Moreover, for BIRCH vs DBSCAN and BIRCH vs Spectral, BIRCH seems to be worst performer as indicated by negative comparative deviation, which means that whenever BIRCH has disagreement with DBSCAN and Spectral, the decisions of BIRCH always turnouts to be wrong. KMeans

TABLE 3: NMI, ARI and AMI scores obtained for clustering algorithms

Algos	BIRCH			DBSCAN			KMeans			Spectral		
Data	NMI	ARI	AMI	NMI	ARI	AMI	NMI	ARI	AMI	NMI	ARI	AMI
NC	0.0045	0.0055	0.0040	1	1	1	0	-0.0007	-0.0005	1	1	1
NM	0.5994	0.6505	0.5992	1	1	1	0.3886	0.4878	0.3883	1	1	1
NS	0.6321	0.5656	0.6316	0.9551	0.9749	0.9549	0.6279	0.6149	0.6275	0.9444	0.9588	0.9443



BIRCH	DBSCAN Right	DBSCAN Wrong	
Right	565231	0	Noisy Circles
Wrong	559019	0	
Right	927775	0	Noisy Moons
Wrong	196475	0	
Right	887807	10952	Aniso
Wrong	224034	1457	

(a) BIRCH vs DBSCAN

BIRCH	KMeans Right	KMeans Wrong	
Right	282661	282570	Noisy Circles
Wrong	279089	279930	
Right	799881	127894	Noisy Moons
Wrong	36445	160030	
Right	822470	76289	Aniso
Wrong	109436	116055	

(b) BIRCH vs KMeans

BIRCH	Spectral Right	Spectral Wrong	
Right	565231	0	Noisy Circles
Wrong	559019	0	
Right	927775	0	Noisy Moons
Wrong	196475	0	
Right	892168	6591	Aniso
Wrong	211503	13988	

(c) BIRCH vs Spectral

Fig. 3: Prayatul Matrices of BIRCH with three other clustering algorithms

also shows the similar performance as indicated by negative comparative deviation for KMeans vs DBSCAN and KMeans vs Spectral. However, as noted during visual inspection for Aniso dataset, all four algorithms misclustered some instances, the measures comparative rightness, effective rightness and effective superiority clearly indicate the same. Overall DBSCAN and Spectral are better performing algorithms over BIRCH and KMeans.

The confusion matrix based measure values accuracy, precision and recall as shown in Table 2 clearly indicate that performance of BIRCH and KMeans is poor. However, as noted earlier, these measures failed to indicate that both DBSCAN and Spectral are best performing algorithms across all three datasets. This happens because these measures are incapable of handling arbitrary cluster labels, which is clearly visible for Noisy Moons dataset as both DBSCAN and Spectral produces exactly same clusters as ground truth but with opposite labels. On the other hand, though NMI, ARI and AMI values presented in Table 3 clearly indicate the poor performance of both BIRCH and KMeans. However, these values certainly cannot tell us that the almost 50% of the instances in Noisy Circle dataset are clustered correctly for both BIRCH and KMeans, when compared to DBSCAN and Spectral. In contrast, polarization values clearly indicate this fact. Similarly, polarization value also clearly indicate that more than 50% of the instances in Noisy Moons and Aniso datasets clustered correctly for both BIRCH and KMeans, when compared to DBSCAN and Spectral. It is nearly impossible to get this important insight with the NMI, ARI and AMI values. The proposed comparative deviation measure can give another important insight about algorithms. For instance, KMeans is better than BIRCH in Noisy Circles and Noisy Moons dataset, which is clearly indicated by comparative rightness, effective rightness and effective superiority as well as NMI, ARI and AMI. However, the NMI, ARI and AMI values or confusion matrix based measures certainly cannot tell us that KMeans is incapable of taking right decisions on certain instances while BIRCH can do as indicated by positive comparative deviation values for BIRCH vs KMeans.

4.2.2 *Instance Level Analysis*

Since the prayatul matrix elements are the counts of instance level comparison (instance pairs) of two clustering algorithms and the proposed measures are designed based on it, so the measure values give instance level comparative performance of the two clustering algorithms. To reaffirm this, the indications noted above are analyzed with the prayatul matrices obtained for BIRCH in comparison to other three clustering algorithms as presented in Fig. 3. Clearly, prayatul matrices for clustering algorithms show that BIRCH has high WR values, which means that large number of instance pairs are assigned to wrong clusters but the same has been assigned to right clusters by others, in particular DBSCAN and Spectral. This reaffirm worst performance of BIRCH in comparison to DBSCAN and Spectral. However, BIRCH mostly having more right decision count in comparison to KMeans as RW count is more than WR count but difference is low. This reaffirms the indications in measure based analysis as well as visual analysis that BIRCH and KMeans performance almost same. Moreover, RW values are 0 for both DBSCAN and Spectral in both Noisy Circles and Noisy Moons, which means that no instance pair has been wrongly clustered by DBSCAN or Spectral that are rightly clustered by BIRCH. At the same time, WR values are very large, which means that many instance pairs that are wrongly clustered by BIRCH are rightly clustered by both DBSCAN and Spectral. Negative comparative deviation values are clear indicative of this fact.

5 CONCLUSION

In this paper, an alternative approach is proposed for direct comparison of clustering algorithms at instance level of test datasets. A direct comparison matrix called *Prayatul Matrix* is prepared to account pairwise instance level comparison of two clustering algorithms. Five measures are designed based on the elements of prayatul matrix, which include comparative deviation, polarization, comparative rightness, effective rightness and effective superiority. Results on widely used standard datasets showed that these measures can give some important insight about the outcomes of algorithms by comparing those directly at instance level. Also, these measures are equally capable of indicating the right decisions of algorithms as conventional measures. For instance, two clustering algorithms having same NMI, ARI or AMI value doesn't mean that their outcomes are same at instance level, comparative deviation and polarization give indications on such differences. While comparative rightness, effective rightness and effective superiority measures give the indication for right decisions of comparing algorithms. Moreover, interpretation of the measures is simple, the rule-of-thumbs for end-users is highly positive values imply good performance and negative implies bad performance of primary algorithm.

ACKNOWLEDGMENTS

This work is supported by the Science and Engineering Board (SERB), Department of Science and Technology (DST) of the Government of India under Grant No. ECR/2018/000204.

REFERENCES

[1] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information processing & management*, vol. 45, no. 4, pp. 427–437, 2009.

[2] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[3] S. Pouyanfar, S. Sadiq, Y. Yan, H. Tian, Y. Tao, M. P. Reyes, M.-L. Shyu, S.-C. Chen, and S. S. Iyengar, "A survey on deep learning: Algorithms, techniques, and applications," *ACM Comput. Surv.*, vol. 51, no. 5, sep 2018. [Online]. Available: https://doi.org/10.1145/3234150

[4] M. Fatima, M. Pasha *et al.*, "Survey of machine learning algorithms for disease diagnostic," *Journal of Intelligent Learning Systems and Applications*, vol. 9, no. 01, p. 1, 2017.

[5] J. Qiu, Q. Wu, G. Ding, Y. Xu, and S. Feng, "A survey of machine learning for big data processing," *EURASIP Journal on Advances in Signal Processing*, vol. 2016, no. 1, pp. 1–16, 2016.

[6] C. Zhang, C. Liu, X. Zhang, and G. Almpanidis, "An up-to-date comparison of state-of-the-art classification algorithms," *Expert Systems with Applications*, vol. 82, pp. 128–150, 2017.

[7] R. Xu and D. Wunsch, "Survey of clustering algorithms," *IEEE Transactions on neural networks*, vol. 16, no. 3, pp. 645–678, 2005.

[8] C. Ferri, J. Hernández-Orallo, and R. Modroiu, "An experimental comparison of performance measures for classification," *Pattern Recognition Letters*, vol. 30, no. 1, pp. 27–38, 2009.

[9] S. E. Schaeffer, "Graph clustering," *Computer science review*, vol. 1, no. 1, pp. 27–64, 2007.

[10] A. Biswas and B. Biswas, "Defining quality metrics for graph clustering evaluation," *Expert Systems with Applications*, vol. 71, pp. 1–17, 2017.

[11] K. Pearson, *On the theory of contingency and its relation to association and normal correlation*. Dulau and Company London, UK, 1904, vol. 1.

[12] J. T. Townsend, "Theoretical analysis of an alphabetic confusion matrix," *Perception & Psychophysics*, vol. 9, no. 1, pp. 40–50, 1971.

[13] L. Hubert and P. Arabie, "Comparing partitions," *Journal of classification*, vol. 2, no. 1, pp. 193–218, 1985.

[14] D. Steinley, "Properties of the hubert-arable adjusted rand index." *Psychological methods*, vol. 9, no. 3, p. 386, 2004.

[15] T. O. Kvålseth, "On normalized mutual information: measure derivations and properties," *Entropy*, vol. 19, no. 11, p. 631, 2017.

[16] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," *The Journal of Machine Learning Research*, vol. 11, pp. 2837–2854, 2010.

[17] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural computation*, vol. 10, no. 7, pp. 1895–1923, 1998.

[18] B. Everitt, "r× c contingency tables," in *The analysis of contingency tables*. Springer, 1977, pp. 38–66.

[19] J. Xu, Y. Zhang, and D. Miao, "Three-way confusion matrix for classification: A measure driven view," *Information sciences*, vol. 507, pp. 772–794, 2020.

[20] X. Deng, Q. Liu, Y. Deng, and S. Mahadevan, "An improved method to construct basic probability assignment based on the confusion matrix for classification problem," *Information Sciences*, vol. 340, pp. 250–261, 2016.

[21] R. Susmaga, "Confusion matrix visualization," in *Intelligent information processing and web mining*. Springer, 2004, pp. 107–116.

[22] E. Beauxis-Aussalet and L. Hardman, "Simplifying the visualization of confusion matrix," in *26th Benelux Conference on Artificial Intelligence (BNAIC)*, 2014.

[23] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[24] F. Pedregosa *et al.*, "Comparing different clustering algorithms on toy datasets - scikit-learn," https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html, accessed: 5-08-2022.

Dr. Anupam Biswas is currently working as an Assistant Professor in the Department of Computer Science and Engineering, National Institute of Technology Silchar, India. He has received Ph.D. degree from IIT (BHU), Varanasi, India. He has published several research papers in transactions, reputed international journals, conference and book chapters. His research interests include Machine learning, Social networks and Evolutionary computation. He has six patents, out of with three are granted Germany patents and one is granted South African patent. He is the Principal Investigator of two on-going DST-SERB and Co-investigator of two other sponsored research projects in the domain of machine learning and evolutionary computation. He has served as Program Chair of BigDML 2019 and Publicity Chair of BigDML 2021. He has served as General Chair of 25th FRSM 2020. He has edited eight books that are published by various series of Springer and Elsevier.