International Research Journal of Mathematics, Engineering and IT

ISSN: (2348-9766)
Association of Academic Researchers and Faculties (AARF)
Impact Factor- 5.489 Volume 6, Issue 3, March 2019
Website- www.aarf.asia, Email : editor@aarf.asia , editoraarf@gmail.com

Efficient K-Means Clustering for Remote Sensing and Handwritten Digit Recognition

Ms. Suman Choudhary
Research Scholar
Deptt of Computer Science and Applications,
Maharishi Arvind University,Jaipur,Rajasthan,India

Prof(Dr.)Mahaveer Kumar Sain
Professor & Dean
Deptt of Computer Science and Applications,
Maharishi Arvind University,Jaipur,Rajasthan,India

Abstract*—Clustering is a powerful unsupervised machine learning technique for grouping similar occurrences together. This Clustering is primarily used to generate clusters of high quality, enabling the discovery of hidden patterns and information inside massive datasets. It has extensive applications in several fields, including medical, gene expression, image processing, healthcare, agriculture, image processing, fraud detection, and profitability analysis, among others.The focus of this study is on the advantages and disadvantages of partitioned clustering, as well as hierarchical clustering. Clustering can also provide valuable insights into the structure and relationships within the data and can be used to improve decision-making in various domainsOne of the most utilized clustering algorithms, k-means divides a dataset into k groups depending on how similar or distant they are from one another. K-means may be utilized for a wide variety of tasks, including but not limited to picture segmentation, customer segmentation, and anomaly detection. Dimensionality reduction is useful on its own, but it may also be used as a preprocessing step for other machine learning methods like classification and regression the complete dataset is taken into account all at once, and the cluster centroids are then determined. With huge datasets, this may be prohibitively computationally costly. Mini-batch K-means was selected because it is quicker at convergence and has lower processing costs than the standard k-means method. In this study, both models were used for performing clustering operations on the two datasets: UCI repository of forest cover type and Kaggle MNIST dataset for handwriting recognition. For the first dataset, K-means shows higher Silhouette and Calinski-Harabasz scores of 0.2637 and 2726.41 and a lower Davies-Bouldin score of 1.4735. For the second dataset, K-means shows higher homogeneity, accuracy, Calinski-Harabasz, Davies-Bouldin scores of 0.8628, 0.9025, 202, 121.16, respectively, and lower inertia, completeness, and v score 233279.56, 0.3627, 0.5107, respectively.

Keywords*—Clustering, clustering techniques, UCI respiratory for ML, MNIST dataset, data preprocessing.

I. INTRODUCTION

The literature provides a wide variety of clustering definitions, ranging from simple to complex. Everyone agrees on the most basic definition, which centers on the act of clustering data that is otherwise related. Cluster analysis groups object that have similar characteristics based on their positions in a multidimensional space or a vector of measures. In contrast to one another, clustering (unsupervised classification) and discriminant analysis both aim to solve the same issue in different ways (supervised classification). After being shown a series of labeled samples, supervised classification attempts to give a label to an unlabeled pattern [1]. It is common practice to acquire class descriptions by observing labeled (training) patterns and then applying those descriptions to a new pattern in order to assign it a label. The goal of clustering is to find meaningful groupings among large collections of unlabeled patterns. Labels are assigned to clusters, but the categories themselves are "data-driven," meaning they are determined solely by examination of the data[1].

To better understand how your data is organized, consider using cluster analysis either on its own or as a first step before applying any further data mining methods to the found clusters. Several clustering algorithms, including K-nearest neighbors and K-means as well as density-based and SSC-EA-based approaches, have been developed and classified according to various criteria. It's also possible to have a category or numerical data collection. The distance function between data points may be defined intuitively by capitalizing on the inherent geometric features of numerical data. Categorical information may originate from numerical data, such as counts, or qualitative data, such as interviews[1].

A. Clustering Analysis Techniques

Cluster analysis is a data mining task since it is used for things like searching, making suggestions, and organising data. Clustering techniques group datasets with similar properties together. Clustering is an unsupervised learning method that contrasts with categorization by grouping objects in a collection into subsets defined by their similarities rather than their differences. This produces meaningful clusters in which items inside a given cluster are quite diverse from one another while objects pertaining to different cluster groups are very alike to one another. After the clustering procedure has been completed, only then will the clusters be recognized. Cluster analysis classifies information according to its relevance or utility (clusters). The "natural" data structure must be reflected in the created clusters if relevant clusters are the aim. Cluster analysis may be expanded upon in several ways, such as for data compression and for quickly discovering neighboring places. Cluster analysis has been used in many different areas, including the social and behavioral sciences, computing, information retrieval, statistics, ML (machine learning,) pattern recognition, and data mining. In this section, you'll learn the basics of cluster analysis. Briefly, they explain a modern approach that makes use of a concept-based approach. The "dimensionality curse" is a factor that must be

considered in every clustering method used for high-dimensional data." [2][3][4][5].

B. Types of Clustering techniques

Clustering may be accomplished in a variety of ways; every clustering approach yields distinct cluster kinds. Some need input user parameters, such as number clusters to be generated, whilst others determine the data type and quantity. The most significant advancements have been advent of density-& grid-based clustering techniques. Five different clustering algorithm classes may be distinguished:

- Model-based technique;
- Partitioning technique;
- Model-based technique;
- Hierarchical technique;
- Density-based technique

1) Model-based technique

Certain data and "mathematical models" are made more suitable by this technique. In comparing standard clusters that define groupings of items, the "model-based clustering approach" as well provides features for every group in which every group identifies a notion or category. "Induction methods," "decision trees," and "neural networks" are often used examples of such techniques. Each node in a "Decision Tree" represents some piece of information—in this case, a "perception and a probabilistic" explanation for a potential action. For each group, the "Neural Networks algorithm" assigns a "neuron" or prototype. The "neurons" assigned to the provided data will correlate with the neurons in the prototype. During the learning phase, each such connection is analyzed to determine its impact. The "self-organizing map (SOM)" is now one of the most used "neural algorithms" for clustering[7].

2) Density-based Clustering

Clustering using partitions is similar to the density-based method. Density-based clustering separates the low-density area from the high-density area. Dense parts that are linked together and expand in any direction are considered to be clusters. This enables the identification of different types of clusters using density-based approaches and provides a natural safeguard against outliers. Clustering techniques that rely on density measurements may be roughly categorized into two subtypes: density-based connectivity and density function. Measured by the local distribution of closest neighbors, density and connectedness are two key characteristics in density-based relationships. Density-based connection algorithms like DBSCAN and GDBSCAN and optical flow algorithms like OPTICS and DBCLASD and density function implementations like DENCLUE are all available [6].

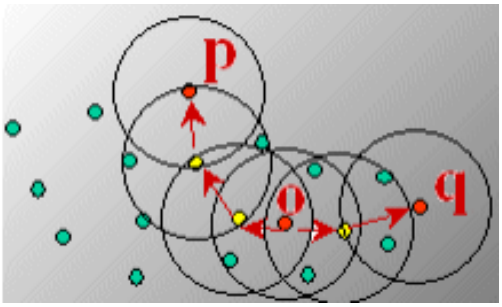

Fig. 1. Density-based Clustering

3) Partitioning-based clustering algorithms

Combinatorial optimization techniques, often known as iterative relocation algorithms, are perhaps the most used kind of clustering algorithm. These algorithms divide data in a way that minimizes a predefined clustering criterion by repeatedly moving data points between groups. A basic iterative approach, such as K-medoids or K-means, cannot guarantee global optimal solution due to the local nature of convergence. The issue of local minima might be eliminated by employing exhaustive search techniques, since the no. of points in every data collection is always limited and, by extension, so is the number of unique partitions. Nevertheless, this is only true in principle, since it is known that determining the globally optimum partition is an NP-hard task and exhaustive techniques are not practical. The number of ways in which n observations may be split into K subsets is a Stirling number of the second type, defined as

$$S_n^{(K)} = \frac{1}{k!}\sum_{i=0}^{i=K}(-1)^{K-i}(Ki)i^n$$

This demonstrates that it is difficult to enumerate all potential divisions, even for comparatively simple issues. The issue becomes much more difficult whenever the no. of clusters is unexlained as well. The total no. of permutations is therefore the sum of second-kind Stirling numbers:

$$\sum_{i=1}^{i=K_{max}} S_N^{(I)}$$

where Kmax is the maximum no. of cluster and it is obvious that Kmax<= n. Even with state-of-the-art computer equipment, exhaustive search techniques are still too time-consuming. The quantity of data and computing power have both grown exponentially over the last several years, making it appear like an endless race. Thus, iterative optimization is preferable than exhaustive search [7].

4) Hierarchical technique

Recursive partitioning is the foundation of hierarchical clustering, whereby datasets are recursively partitioned into finer granularity or higher granularity in order to make the cluster. Each of the nodes outside of the tree's internal node represents a data point, often known as a sibling of the parent cluster, while the tree's core node represents the cluster itself. This grouping on a given collection of data points captures meaningful information. As an example, social networks may be used to build clusters of related groups, and digital photographs can be segmented repeatedly into discrete regions with increasing granularity[8]. Having a high number

of partitions, each of which is connected to a level, is one of the primary attractions of hierarchical clustering. This is a data structure known as a binary tree. This clustering approach is preferable to K-means since it does not need knowing number of clusters in advance[9].

II. LITERATURE REVIEW

This section offers an in-depth discussion of big data challenges, clustering techniques, particularly the KMeans Clustering Algorithm, the MapReduce Framework, and big data applications and authors of various research described below:

Fuzzy logic for cluster selection is a technique proposed by Phommasan, Widyawan, and Mustika (2022) to increase the WSNs' useful lifetime (FLCE). The proposed scheme's experimental results show that, in comparison to the low-energy adaptive clustering hierarchy, the FLCE technique greatly increases network lifetime (LEACH)[10].

The optimal number of clusters was determined using three methods proposed by Patel, Sivaiah, and Patel (2022): the elbow technique, the gap statistics method, and the Silhouette method. The AHC (agglomerative hierarchical clustering) method and K-means techniques were also used to find the optimal k-value for the dataset. Both methods are put to the test on a small dataset, with validation measures including connectivity, Silhouette, and Dunn, utilized to find optimal number of clusters[11].

In this study, Mahmud *et al.*, (2020) improved DBMS subtrajectory clustering, as well as incremental and progressive cluster analysis. Unfortunately, it was not simple to conduct such an operation on massive amounts of data in a centralized manner, hence parallel and distributed algorithms were required. After realizing that the underlying trajectory join query was the bottleneck, they use the MapReduce programming approach to solve the Distributed Subtrajectory Join problem. Eventually, this query was used to solve the scalability and efficiency issues using Distributed Subtrajectory Clustering [12].

Focusing on clustering's real-world applications, Vijaya, Sharma, and Batra (2019) compared several connecting techniques and methods to find out what matters most when combining clusters of different sizes. In this study, they use hierarchical agglomerative clustering as an example of real-time retail data in an effort to draw conclusions. they've also evaluated the variation in results across a number of various criteria, such as "ward," "single linkage," "full linkage," etc[13].

According to Kumar and Singh, (2019), hybrid clustering was proposed as a novel approach to address the shortcomings of previous clustering methods. In terms of precision, speed, and other metrics like recall and F-measure, they contrast unique hybrid approaches to state-of-the-art. The outcome of the experiments validates the superiority of suggested hybrid clustering method in terms of F-measure, recall, and accuracy [14].

The goal, as stated by Alam et al. (2019), was to develop a network-agnostic CH-based HDC (Histogram-based Data Clustering) technique. The HDC groups similar data together into clusters, from which it selects cluster centers to perform in-network data decrease. The suggested HDC was shown to have the potential to significantly minimize duplicate data while outperforming the present gold standard approach based on empirical findings utilizing real-world sensor data[15].

Here, Marlen et al. (2019) describe how big data may be used in the near future in Kazakhstan's smart grids and present an in-depth examination of the field's potential. In addition, a case study of load profiling was presented in this research utilizing the dataset that records residential energy usage in London from November 2011 to February 2014. Here, an unsupervised analysis was performed using the K-means clustering technique. This study has shown how energy analytics may have an effect on the energy sector of Kazakhstan's infrastructure and how some of the difficulties with power systems can be circumvented. Although it's clear that raising the understanding of both government (the actual policy implementers) and people were necessary to achieve all these potential (i.e.ones who would make it possible) [16].

Sarfraz, Sharma, and Stiefelhagen, (2019) provided a novel clustering approach, expressed as a single clustering equation, that can detect clusters in the data on its own. In order to locate huge chains and groups in the data, it was sufficient to just look at the first neighbor of each sample, according to the basic premise. Their approach eliminates the requirement for hyper-parameters, distance criteria, and/or a predetermined number of clusters, in contrast to the majority of currently available clustering methods. An example of a hierarchical agglomerative approach, the presented method fits right in. The method may be used to big practical issues with little to no additional computing burden. Using popular datasets from a variety of fields, the innovative approach outperformed state-of-the-art clustering techniques by a wide margin. These datasets ranged in size from 1077 to 8.1 million samples[17].

III. SIMULATION TOOL

Python's flexibility makes it useful for many different kinds of programming. It was first presented and produced in 1991 by Guido van Rossum. Python produces excellent products in terms of code quality, readability, interaction, and so on. Python was designed to be simple and straightforward. English keywords are used regularly, and punctuation is used by some. It lacks several syntactic structures that are present in other languages.

- **Perceived in Python:** During runtime, Python is executed by the interpreter. Your program doesn't need to be finished until you're ready to launch it. Similarities exist between PERL and PHP.
- **Python allows for dynamic interaction**: You may also write your programs by talking to an interpreter directly in a Python prompt.
- **Object-oriented programming in Python:** Python encourages modules, a kind of encapsulation that is common in object-oriented applications.
- **Software for Novices: Python:** Python is great for novice programmers since it allows them to quickly

and easily create web-based sports-related word processors. [18].

More precisely, Python is a programming language that allows the user to concentrate more quickly to solve domain problems instead of struggling with the complexities of how a machine works in comparison to other programming languages like C, FORTRAN, or Java. By the following characteristics, Python achieves this objective:

- Python is a language of high level, which means it sums up the technical details of computers. For example, Python does not make the users too much worry about computer storage or correct variables definitions and makes sure what the programmer is trying to convey. Python does not. To get closer to English prosthetics or math, a high-level language can also be conveyed. The simple, "small ceremonies" character of Python is suitable for literary programming. Python For artificial and statistical analyses, for example, Python can be used. As the UCAR scientist described earlier, Python can be used for several heterogeneous workflows.
- Python has the main library but many third-party implementations, which provide a wide range of popular codebases and models of problem-solving.
- Programmers can quickly find solutions and sample code for problems with the help of Google and Stackoverflow.

A. Running Python

Python can be started in three different ways -

1. **Interactive Interpreter:** Python may be launched from any system that has an interpreter command line or shell window, including Unix, DOS, and others. Line python join. You should instantly keep on coding in the interactive interpreter.

TABLE I. NOW ARE ALL COMMAND-LINE CHOICES AVAILABLE

Option	Description
-d	Perform debug
-O	Optimized bytecode generation (finding in records with. Pyo)
-S	Do not execute the import website to search for startup Python paths
-V	Comprehensive import monitor result (verbose output)
-X	Unable built-in class (just usage strings) exceptions; obsolete from version 1.6.
-c cmd	Execute the script Python transmit as cmd string
File	Execute Python file script

2. **Script from Command-line:**You can invoke Python interpreters from the command line, as shown in the following statement

Note: Make sure file enables execution in permission mode

3. **Integrated Development Environment:** If you get a GUI framework that supports the Python program, we can also run Python after the GUI environment.

- **UNIX:** IDLE is 1st Python Uni
- x IDE.
- **Windows:** Python Win is Python's first Windows client and GUI IDE.
- **Macintosh:** Python versions of Macintosh and IDLE-IDE are obtainable as MacBinary or BinHex files from the main website.

B. Jupyter

When it comes to interactively developing or observing data science projects, the Jupyter Notebook is the standard tool. A notebook is a document that contains many types of material, such as code and its output, as well as a picture, some prose, some arithmetic, and so on. Notebooks are more likely to be at the center of cutting-edge information science, technology, and innovation because to their intuitive workflow, which promotes iterative yet quick setup. Even worse, there are no costs associated with using the Jupyter open-source project. The IPython Notebook was initially offered as a reference in 2010, and Jupyter is its successor. Although Python is the most popular choice for usage in Jupyter Notebooks, other programming languages are supported.

C. Datatype Conversion

The need to transform data between internal representations occurs often. To tell one type from another, you need just use the corresponding type name. Several in-built functions exist to facilitate the transformation of data kinds. The value transformed is represented by a new object that is restored via these methods. This is a list of the few:

TABLE II. FUNCTIONS & DESCRIPTION

S.no.	Functions & Descriptions
1.	**int (x [, base])**; Conversion of x to integer. basis stipulates that x is number.
2.	**float(x)**; Transforms x into set of floating points.
3.	**complex (real [, imag])**; Produces an interesting no.
4.	**eval(str)**; Returns an object to evaluate a string.
5.	**str(x)**; Conversion of object x to a string file.
6.	**set(s)**; Turns s into set
7.	**tuple(s)**; Turns s into tuple.
8.	**list(s)**; Turns s into list.
9.	**repr(x)**; Turns item x to a string expression.
10.	**dict(d)**; Make dictionary. d shall be tuple (key, value) series [19].

IV. PROBLEM STATEMENT

Despite their importance, efficient clustering techniques for big data face several challenges, including As the volume of data grows, clustering algorithms need to scale to handle the increased data volume. Traditional clustering algorithms

can become inefficient and ineffective when dealing with large datasets. Scalable clustering algorithms are needed to handle big data efficiently. The number of dimensions in big data is often rather high, which can make clustering challenging. Distances between data points tend to grow as the number of dimensions grows, making it harder to find significant clusters. Efficient clustering techniques must be able to handle high-dimensional data effectively. Big data can be noisy, with outliers and irrelevant data points that can skew clustering results. Efficient clustering techniques must be able to handle noisy data and identify meaningful clusters despite noise. Clustering algorithms can produce large volumes of data, which can be difficult to interpret and analyze. Efficient clustering techniques must be able to produce interpretable results, enabling analysts to extract meaningful insights and knowledge from the data. Clustering algorithms can be computationally intensive, requiring significant computational resources. Efficient clustering techniques must optimize resource usage, making it possible to cluster large datasets on limited resources.

Addressing these challenges requires the development of efficient clustering algorithms which can handle large volumes of high-dimensional, noisy data, produce interpretable results, and optimize resource usage. By overcoming these challenges, efficient clustering techniques can enable the analysis of big data, leading to better decision-making, insights, and competitive advantage.

V. Research Methodology

A. Dataset gathering

The process of gathering data from various sources for use in training ML models is sometimes referred to as "dataset collection." The quality and quantity of the data collected greatly affect the final model's accuracy and generalizability. Two datasets were used in this investigation: one from the UCI repository describing various forest cover types, and another from the handwriting recognition site Kaggle.

B. Data Preprocessing

Data preparation, or getting the data ready for use by the ML algorithm, is a crucial stage in the ML process. By fixing frequent problems in the data, data preprocessing helps make ML models more accurate and trustworthy. In the context of ML challenges, it is common knowledge that the stages of data preprocessing and filtering consume a significant amount of processing time. The cleaning, normalization, transformation, feature extraction and selection, and other aspects of data are included in the pre-processing stage. The completed training set is the result of applying preprocessing to the data [20].

1) Checking null values

Data preparation tasks, such as checking for null values, are crucial in getting data ready for ML algorithms. Incomplete or inaccurate data gathering are two common causes of null values, which are also known as missing values. Problems like biased analysis, wrong conclusions, and less accurate ML models may be avoided with proper handling of null values.

2) Checking duplicate value

Data preparation includes checking for duplicate values, which is crucial for data preparation for ML algorithms. Duplicate values can occur due to various reasons such as errors in data entry or data duplication. Duplicate values can negatively impact the accuracy and reliability of ML models, leading to biased or incorrect analysis.

3) Categorical encoding

Accurate analysis requires a firm grasp of the underlying facts. The data is preprocessed to help algorithms and boost efficiency before the real analysis begins. Figure 2 depicts several more frequent statistical classification schemes. It makes no sense to calculate statistical measures like the mean or standard deviation with qualitative variables, which are only descriptive or categorical, and it does make sense to do so with quantitative variables, which have a numeric meaning. Figure 2 shows that there are two types of categorical data: nominal (not ordered) and ordinal (with some particular order). A variable measured on a nominal scale has categories that do not lend themselves to being ranked. Blood type, Genotype, eye color, gender, race, zip code, and iris color are all instances of nominal variables. When comparing numbers, ordinal scales emphasize rank rather than absolute value. Ordinal variables are often used in surveys, and responses such as "very dislike," "dislike," "neutral," "like," and "highly like" represent a range of possible responses.[21].

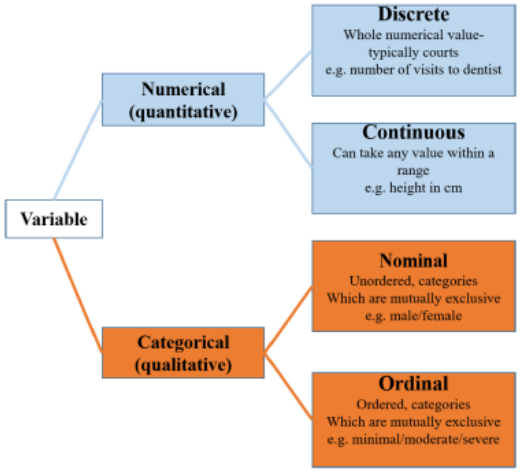

Fig. 2. Types of variable

C. Data Normalization

Normalization of the data is a primary objective in data processing. How well data are normalized has a significant impact on how well ML algorithms function. When data is normalized, each feature's original value is transformed into a specified range. Whether or not this extra step is necessary depends on how sensitive the chosen ML model is to the feature's value[22]. It is a process of transforming data into a common scale to remove differences in the magnitude of variables. The primary objective of data normalization is to standardize the measurement of all variables so that they all contribute equally to the results of the study. This is important because variables that are measured in different units or have different scales can create bias in the analysis.

When applied to features, the Standard Scaler (SS) method reduces the average and normalizes the variance to provide a consistent output. The fact that the normalized value can be calculated quickly and easily using just the mean and variance is only one of its many benefits. In contrast, Standard Scaler favors normally distributed data and is very sensitive to outliers[23].

D. Classification

In this research, mini-batch K-means & K-means are applied to provided UCI and Kaggle datasets. Both methods are explained below.

1) K-Means Clustering

K-means clustering is widely employed in cluster analysis due to its ease of utilization and quick convergence. The K-means method is an unsupervised learning procedure that may be used to cluster data. There are several applications for the K-means clustering technique. It's an easy iterative method for grouping data. The initial centroid is found by computing the distance mean between all points in the dataset, where K is the number of classes. The Euclidean distance, as indicated in (i) is chosen as similarity index, and clustering objectives minimize sum of squares of the different kinds of data points in dataset X, which contains n multidimensional data points and the category K to be separated[24].

$$D = \sqrt{\sum_{j=1}^{NDIM} (x_{ij} - c_{jk})^2} \qquad \text{(i)}$$

where D is Euclidean distance between data point i and centroid k; NDIM—is the number of dimensions; xij—is coordinate in dimension j of data point i; and cjk—is coordinate in dimension j of centroid k.

The K-means clustering technique has six distinct steps. The first step is to settle on a target value for the cluster size (K). The outcome of the algorithm is very sensitive to the choice of K. The second step involves a random selection of centers. In the last step, we measure how far apart each data point is from each of the cluster centers. The Euclidean distance is often utilized in calculations. In the last step, clusters are formed such that the data points have a minimal centroid distance. In the fifth step, we do a new center calculation by (ii). Ultimately, the algorithm goes back to the third step if the centroid has been modified, else it ends[25].

$$C_{jl} = \frac{1}{N_l} \sum_{i=1}^{N_l} x_{ij} \qquad \text{(ii)}$$

where, xij—is coordinate in dimension j of data point i; Nl—is number of data points in cluster l; and Cjl—is centroid in dimension j of cluster.

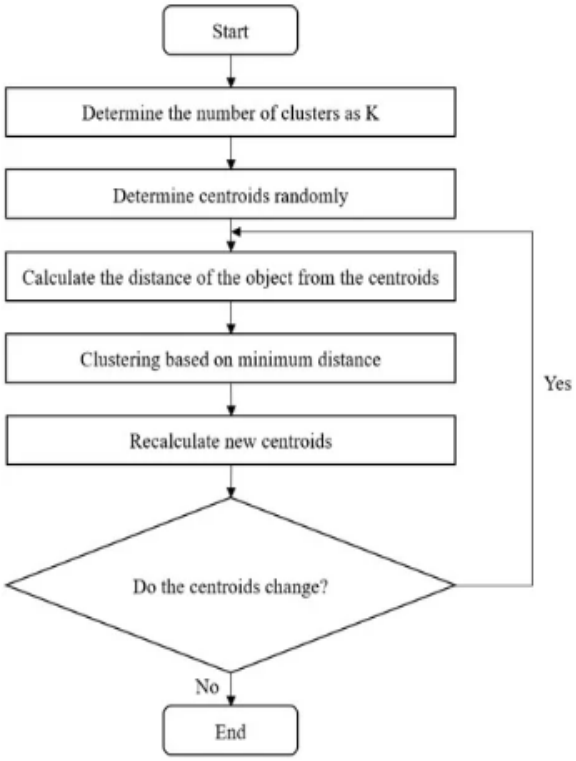

Fig. 3. K-means Clustering algorithm

2) Mini-Batch K-means

The k-means method has been tweaked to create the tiny batch k-means clustering technique. Mini-batches are used to speed up calculations on massive datasets. In addition, it tries to maximize clustering outcome. Mini-batches, or random subsets of whole dataset, are used as input by mini batch k-means algorithm to accomplish this goal. Mini batch k-means is regarded to be quicker than k-means and is often utilized for big datasets. [26]. Mini-batch k-means algorithm hyper-parameters include batch number & size of iterations. The number of iterations controls how often the centroids are updated by the algorithm, and the batch size specifies how many data points are handled in each iteration. Mini-batch k-means outperforms traditional k-means algorithms for large datasets since it requires less time and resources to converge.

VI. Results Illustration

A. Dataset Description

In this study, two datasets have been used: first is the UCI repository for ML and second is the MNIST dataset.

The first data set was built using resources found in the UCI Machine-Learning repository. Types of forest cover prediction using just cartographic factors (no remotely sensed data). Utilizing data from Region 2 RIS (Resource Information System) of USFS (US Forest Service), actual forest cover category for a particular observation (30 x 30 meter cell) was determined. The independent variables were constructed using information that was first collected by the US Geological Survey (USGS) and the US Forest Service (USFS). The qualitative independent variables are represented as binary (0 or 1) columns in the raw (unscaled) data (wilderness areas and soil types).

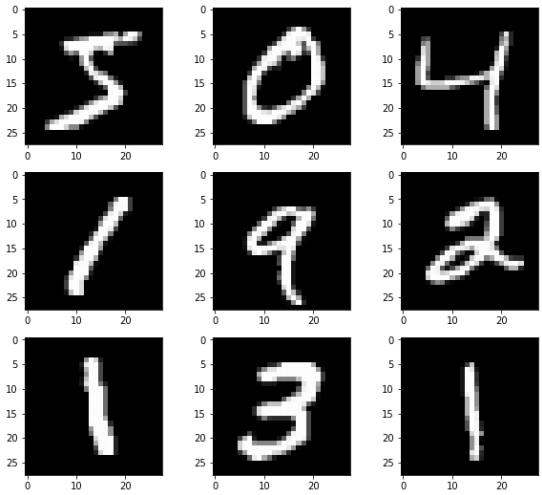

Fig. 4. Sample of the Kaggle handwritten dataset

B. Performance Matrix

Performance evaluation is an essential step in ML to assess how well a model is able to make predictions on new, unseen data. The efficacy of a ML model may be evaluated in a variety of ways, such as:

1) Silhouette score

Utilizing silhouette score function of scikit-learn package, the mean silhouette coefficient is computed. The silhouette coefficient is calculated by taking mean value of intra-cluster distance and mean value of the nearest-cluster distance for every data point [27]. The silhouette coefficient for a sample is

$$S_{i=2\ to\ n} = (S_i - S'_i)/Max(S, S''_i)$$

where, S′i = Average distance between *i*th cluster with different groups/clusters. Si = Average distance of items between *i*th group/cluster. Max (Si,S′i) = Average distance between Si with S′i.

- A data point is properly clustered if its silhouette score is near + 1.
- The clustering of data points is incorrect if its silhouette score is close to -1.
- The silhouette score being close to 0 suggests that data point was incorrectly labeled.

2) Calinski-Harabasz score

A greater value for Calinski-Harabasz index, also called as Variance Ratio Criterion, is the ratio of total dispersion among clusters to indicate superior performance. It's a measure of how dissimilar individual clusters are from one another. The conventional definition of a cluster assumes that clusters are dense and spatially distinct, hence a larger number suggests this[28]. The formula for calculating the score is:

$$CH = (B / (k - 1)) / (W / (n - k))$$

where B is between-cluster dispersion, CH is Calinski-Harabasz index,W is within-cluster dispersion, k is number of clusters, and n is the total number of data points.

3) Davies-Bouldin Score

Davies-Bouldin score is utilized to assess the reliability of a clustering method. It is the ratio of average similarity between two clusters to the average dissimilarity among 2 clusters. The index of a database is an internal rating system. The greater clustering tolerance, the lesser value of database index. There is, however, a disadvantage to this. Despite its obvious benefits, this method may not be the best choice for data retrieval[29].

$$DB = \frac{1}{k}\sum_{i=1}^{k} max\frac{s_i + s_j}{d_{ij}}$$

where si is the average distance between all objects in cluster *i* and cluster *i* centroid, *k* is the cluster number, dij is distance between *ith* and *j th* cluster centroids

4) Inertia

In clustering algorithms, inertia is a performance assessment parameter utilized to measure accuracy of resulting clusters. The inertia is calculated by adding squares of lengths from every sample to middle of the collection. The clustering performance improves with decreasing inertia.

5) Homogeneity

The homogeneity of a set of clustering results is a useful measure of the effectiveness of a clustering method. It evaluates how strictly one kind of data is represented in each cluster. A perfect clustering with respect to homogeneity would have all clusters containing only samples from a single class. Better clustering outcomes may be achieved with a higher homogeneity score.

6) Accuracy

The classification process's accuracy is based on the proportion of correct to wrong predictions. The classification accuracy can be computed using an equation. (1)

$$\frac{TP + TN}{P + N} = \frac{TP + TN}{TP + TN + FP + FN}$$

C. Comparative Results

TABLE III. COMPARISON OF PERFORMANCE METRICS UCI REPOSITORY FOR ML DATASET

Model	Silhouette Score	Calinski-Harabasz Score	Davies-Bouldin score
K-means	0.2637	2726.41	1.4735
Mini-batch K-means	0.2180	2615.90	1.6175

Table 3 shows the comparison of performance evaluation scores of both mini-batch k-means & k-means. K-means shows higher Silhouette and Calinski-Harabasz scores and lower Davies-Bouldin scores.

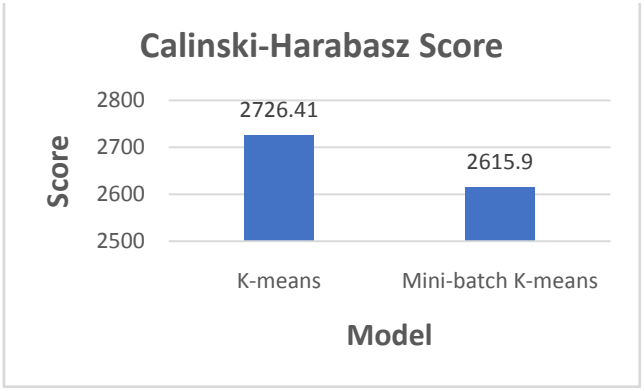

Fig. 5. Calinski-Harabasz score comparison

Calinski-Harabasz scores for k-means and mini-batch k-means models are shown side-by-side in Figure 5. K-means outperforms mini-batch k-means with a score of 2726.41. Hence, k-means shows better Calinski-Harabasz score.

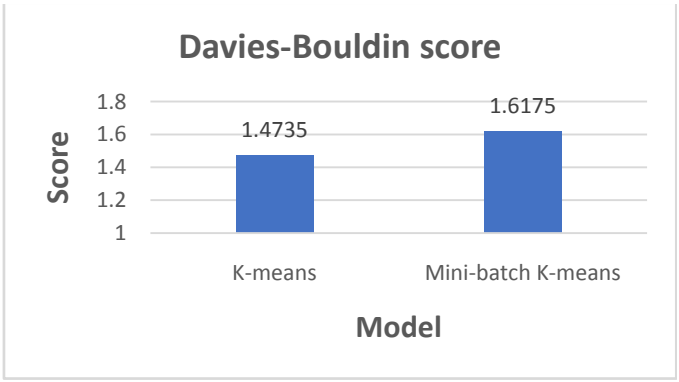

Fig. 6. Davies-Bouldin score comparison

Figure 6 shows the comparison of Davies-Bouldin score of both k-means & mini-batch k-means model. K-means shows a lower score of 1.4735 than mini-batch k-means. Hence, k-means shows better Davies-Bouldin score.

Overall, the K-means model shows better performance when comparing performance metrics shown above.

TABLE IV. COMPARISON OF PERFORMANCE METRICS MNIST DATASET

Model	Inertia	Homogenes	Accuracy	Calinski-Harabasz Score	Davies-Bouldin score	Completeness Score	V Score
K-means	233279.56	0.8628	0.9025	202	121.16	0.3627	0.5107
Mini-batch k-means	239491.68	0.8601	0.8935	191.74	118.59	0.3678	0.5153

Table 4 shows the comparison of performance metrics of both k-means and mini-batch k-means. K-means shows higher homogeneity, accuracy, Calinski-Harabasz, and Davies-Bouldin score and lower inertia, completeness, and v score.

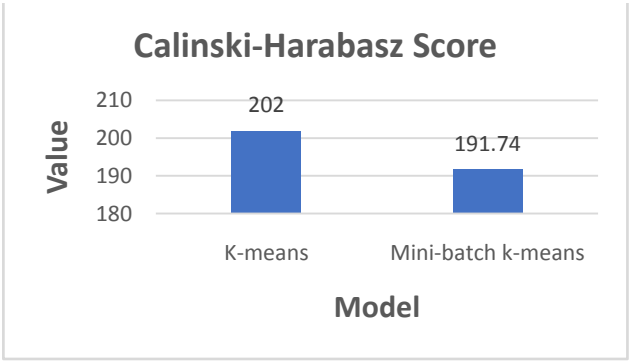

Fig. 7. Calinski-Harabasz score comparison of K-means and Mini-batch K-means

Figure 7 shows the comparison of Calinski-Harabasz score of both models. K-means shows a higher Calinski-Harabasz score of around 202 as compared to mini-batch k-means.

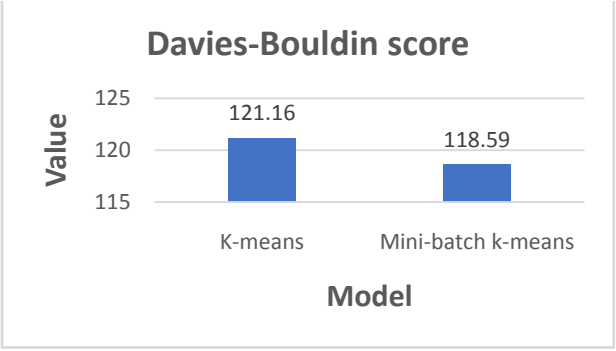

Fig. 8. Davies-Bouldin score comparison of K-means and Mini-batch K-means

Figure 8 shows the comparison of Davies-Bouldin score of both models. K-means show a higher Davies-Bouldin score of 121.16 as compared to mini-batch k-means.

VII. CONCLUSION

Clustering methods have been widely adopted across disciplines for their ability to reveal hidden patterns and group together seemingly unrelated data items. This study investigates the usefulness of clustering methods in two unique applications: handwriting recognition and forest cover type detection.

K-means shows higher Silhouette and Calinski-Harabasz scores of 0.2637 and 2726.41 and a lower Davies-Bouldin score of 1.4735. The results demonstrate that K-means can effectively identify forest cover types and provide valuable insights into forest management and conservation. For handwriting recognition, we used the MNIST dataset was used. The grouping process was carried out using the K-means and Mini-batch K-means clustering methods. Utilizing inertia, uniformity, precision, completeness score, calinski-harabaszscore, profile score, V score, and davies-bouldin score, performances oftwo models were assessed and contrasted. According to the findings, K-means performs better in terms of homogeneity, precision, Calinski-Harabasz score, and Davies-Bouldin score (0.8628, 0.9025, 202, and 121.16, respectively), while performing worse in

terms of inertia, completeness, and v score (33279.56, 0.3627, and 0.5107, respectively).

REFERENCES

[1] A. N. Gupta and P. Arti, "A Review : Study of Various Clustering Techniques in Web Usage Mining," vol. 3, no. 3, pp. 5888–5890, 2014.

[2] C. C. Aggarwal and C. Zhai, *A survey of text clustering algorithms. In Mining text data*. 2012.

[3] A. Fahad *et al.*, "A survey of clustering algorithms for big data: Taxonomy and empirical analysis," *IEEE Trans. Emerg. Top. Comput.*, 2014, doi: 10.1109/TETC.2014.2330519.

[4] A. Hatamlou, "In search of optimal centroids on data clustering using a binary search algorithm," *Pattern Recognit. Lett.*, 2012, doi: 10.1016/j.patrec.2012.06.008.

[5] D. Pandove and S. Goel, "A comprehensive study on clustering approaches for big data mining," 2015. doi: 10.1109/ECS.2015.7124801.

[6] P. Bhattacharjee and P. Mitra, "A survey of density based clustering algorithms," *Frontiers of Computer Science*. 2021. doi: 10.1007/s11704-019-9059-3.

[7] S. Ayram and T. Kainen, *Introduction to partitioning-based clustering methods with a robust example*. 2006.

[8] V. Cohen-Addad, V. Kanade, F. Mallmann-Trenn, and C. Mathieu, "Hierarchical clustering: Objective functions and algorithms," *J. ACM*, 2019, doi: 10.1145/3321386.

[9] F. Murtagh and P. Contreras, "Algorithms for hierarchical clustering: an overview, II," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*. 2017. doi: 10.1002/widm.1219.

[10] S. Phommasan, Widyawan, and I. W. Mustika, "Cluster Selection Technique with Fuzzy Logic-based Wireless Sensor Network to increase the lifetime of networks," in *2022 5th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, 2022, pp. 40–46. doi: 10.1109/ISRITI56927.2022.10052871.

[11] P. Patel, B. Sivaiah, and R. Patel, "Approaches for finding Optimal Number of Clusters using K-Means and Agglomerative Hierarchical Clustering Techniques," in *2022 International Conference on Intelligent Controller and Computing for Smart Power (ICICCSP)*, 2022, pp. 1–6. doi: 10.1109/ICICCSP53532.2022.9862439.

[12] M. S. Mahmud, J. Z. Huang, S. Salloum, T. Z. Emara, and K. Sadatdiynov, "A survey of data partitioning and sampling methods to support big data analysis," *Big Data Min. Anal.*, vol. 3, no. 2, pp. 85–101, 2020, doi: 10.26599/BDMA.2019.9020015.

[13] Vijaya, S. Sharma, and N. Batra, "Comparative Study of Single Linkage, Complete Linkage, and Ward Method of Agglomerative Clustering," in *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, 2019, pp. 568–573. doi: 10.1109/COMITCon.2019.8862232.

[14] S. Kumar and M. Singh, "A novel clustering technique for efficient clustering of big data in hadoop ecosystem," *Big Data Min. Anal.*, vol. 2, no. 4, pp. 240–247, 2019, doi: 10.26599/BDMA.2018.9020037.

[15] M. K. Alam, A. A. Aziz, S. A. Latif, and A. Awang, "Data Clustering Technique for In-Network Data Reduction in Wireless Sensor Network," in *2019 IEEE Student Conference on Research and Development (SCOReD)*, 2019, pp. 317–322. doi: 10.1109/SCORED.2019.8896244.

[16] A. Marlen, A. Maxim, I. A. Ukaegbu, and H. S. V. S. Kumar Nunna, "Application of Big Data in Smart Grids: Energy Analytics," in *2019 21st International Conference on Advanced Communication Technology (ICACT)*, 2019, pp. 402–407. doi: 10.23919/ICACT.2019.8701973.

[17] S. Sarfraz, V. Sharma, and R. Stiefelhagen, "Efficient Parameter-Free Clustering Using First Neighbor Relations," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 8926–8935. doi: 10.1109/CVPR.2019.00914.

[18] Tutorials Point (I) Pvt. Ltd., "About the Tutorial Copyright &Disclaimer," *Tutorials Point*, pp. 1–13, 2017.

[19] "Python 3 - Variable Types."

[20] S. B. Kotsiantis and D. Kanellopoulos, "Data preprocessing for supervised leaning," *Int. J. ...*, vol. 1, no. 2, pp. 1–7, 2006, doi: 10.1080/02331931003692557.

[21] M. K. Dahouda and I. Joe, "A Deep-Learned Embedding Technique for Categorical Features Encoding," *IEEE Access*, 2021, doi: 10.1109/ACCESS.2021.3104357.

[22] I. Izonin, R. Tkachenko, N. Shakhovska, B. Ilchyshyn, and K. K. Singh, "A Two-Step Data Normalization Approach for Improving Classification Accuracy in the Medical Diagnosis Domain," *Mathematics*, vol. 10, no. 11, 2022, doi: 10.3390/math10111942.

[23] P. Ferreira, D. C. Le, and N. Zincir-Heywood, "Exploring Feature Normalization and Temporal Information for Machine Learning Based Insider Threat Detection," in *2019 15th International Conference on Network and Service Management (CNSM)*, 2019, pp. 1–7. doi: 10.23919/CNSM46954.2019.9012708.

[24] C. Yuan and H. Yang, "Research on K-Value Selection Method of K-Means Clustering Algorithm," *J*, vol. 2, no. 2, pp. 226–235, 2019, doi: 10.3390/j2020016.

[25] J. Park and M. Choi, "A K-Means Clustering Algorithm to Determine Representative Operational Profiles of a Ship Using AIS Data," *J. Mar. Sci. Eng.*, vol. 10, no. 9, 2022, doi: 10.3390/jmse10091245.

[26] A. Feizollah, N. B. Anuar, R. Salleh, and F. Amalina, "Comparative study of k-means and mini batch k-means clustering algorithms in android malware detection using network traffic analysis," in *2014 International Symposium on Biometrics and Security Technologies (ISBAST)*, 2014, pp. 193–197. doi: 10.1109/ISBAST.2014.7013120.

[27] W. H. S. . Gunarathne, K. D. M. Perera, and K. A. D. C. . Kahandawaarachchi, "Performance Evaluation on Machine Learning Classification Techniques for Disease Classification and Forecasting through Data Analytics for Chronic Kidney Disease (CKD)," in *2017 IEEE 17th International Conference on Bioinformatics and Bioengineering (BIBE)*, 2017, pp. 291–296. doi: 10.1109/BIBE.2017.00-39.

[28] F. Aksan *et al.*, "Clustering Methods for Power Quality Measurements in Virtual Power Plant," *Energies*, vol. 14, no. 18, 2021, doi: 10.3390/en14185902.

[29] A. Ullah *et al.*, "Customer Analysis Using Machine Learning-Based Classification Algorithms for Effective Segmentation Using Recency, Frequency, Monetary, and Time," *Sensors*, vol. 23, no. 6, 2023, doi: 10.3390/s23063180.