Nanotechnology Perceptions
ISSN 1660-6795
www.nano-ntp.com

K Means Clustering Technique for Big Data Analysis

Rajesh Govind Talekar[1,3] , Avinash Vasantrao Khambayat[2,*]

[1]*Research Scholar, Department of Mathematics, Sandip University, Nashik, 422213, Maharashtra, India*
[2]*Professor, Department of Mathematics, Sandip University, Nashik, 422213, Maharashtra, India*
[3]*Assistant Professor in Mathematics ,Department of Applied Science and Humanities, Pimpri Chinchwad College of Engineering, Nigdi, Maharashtra, India*
[1]*rajesh.talekar@pccoepune.org,* [2]*avinash.khambayat@sandipuniversity.edu.in*

Big data analytics is rapidly becoming a critical area of research in computer science and various industries worldwide, demonstrating significant success across sectors like social media, economy, finance, healthcare, and agriculture. We examine the most commonly used machine learning algorithms for big data analytics, emphasizing their ability to manage the unique challenges posed by big data, such as high velocity, large volume, uncertainty, non-stationary characteristics, and real-time data requirements. With the daily generation of gigabytes of data, traditional machine learning techniques fall short due to the distinctive features of big data. Additionally, conventional storage and processing methods are insufficient for the demands of big data environments. This paper explores the challenges associated with applying traditional unsupervised machine learning techniques to big data analytics and presents potential solutions. Our study investigates traditional clustering techniques within big data analysis, emphasizing Spark-enabled K-means clustering for efficient processing. By integrating Silhouette analysis, optimal cluster configurations are identified, enhancing clustering accuracy for large-scale datasets. The work is validated across multiple datasets, enhancing clustering accuracy for diverse and large-scale data collections. Our work highlights effective strategies such as parallel processing, and the use of GPUs and Spark framework as feasible approach to address the various challenges. The findings and insights provided contribute to the ongoing efforts to develop and improve analytical methods capable of handling the complexities and scale of big data.

Keywords - Big Data analytics, machine learning algorithms, K-Means clustering, Spark Freamwork.

1. INTRODUCTION

With the rapid advancement of technology, the volume of digital data being generated has skyrocketed, presenting a myriad of challenges and opportunities. This data is not only vast but also diverse, often characterized by high velocity and the presence of missing or uncertain values. The concept of big data, as defined by the 5V model encompassing Volume, Variety, Velocity, Veracity, and Value, has found applications across numerous domains, including but not limited to social networking, meteorology, online commerce, and finance [1]. Despite the abundance of data, its true value remains untapped without effective and efficient analysis. Big data analytics, therefore, emerges as a crucial technique for extracting meaningful insights from these vast datasets, facilitating informed decision-making and enhancing various aspects of human life and business operations. Machine learning, a subfield of Artificial Intelligence, plays a pivotal role in this context by enabling systems to learn from data and improve performance over time.

Machine learning techniques, such as Supervised Learning, Unsupervised Learning which offer diverse approaches to data analysis. Supervised learning involves mapping input parameters to corresponding outputs using numerical parameters, with algorithms like Decision Trees and Support Vector Machines [2] being commonly employed for classification and regression tasks. Unsupervised learning, on the other hand, focuses on identifying patterns within data without predefined labels, leveraging techniques like clustering and association. Reinforcement learning operates on a hit-and-trial basis, learning from interaction with the environment to achieve specific goals. As an indispensable component of data mining, clustering algorithms play a crucial role in analyzing Big Data [3]. These algorithms are categorized into density-based, partition-based, hierarchical, and model-based clustering methods.

Each clustering method aims to address the common challenge of grouping individual data points in a manner that maximizes similarity within clusters while minimizing dissimilarity between clusters. Typically, clustering algorithms involve the random selection of initial clusters, followed by iterative optimization until an optimal solution is achieved Dave et al [5]. Clustering finds wide application across various domains; for instance, it is employed in intrusion detection systems to identify anomalous behaviors Othman et al. [6] and extensively utilized in text analysis for document classification Fasheng et al [7]. However, with the exponential growth in data volume driven by modern technologies, traditional clustering methods face challenges in scalability and computational efficiency, making them unsuitable for handling very large datasets. Consequently, these methods fail to meet the demands of contemporary data-intensive applications. To effectively handle Big Data, clustering algorithms must be capable of extracting patterns from unstructured, massive, and heterogeneous datasets.

During our practice exercises for exploring clustering techniques relevant to big data analytics, we have run assorted traditional methods simultaneously, while exploiting some distributed computing capabilities in Apache Spark. Among such approaches, K-means is probably one of the most commonly used algorithms for partitioning a dataset into groups based on either similarity or dissimilarity in their characteristics. By using the holistic framework of Spark, K-means clustering was effectively implemented by taking the advantage of its parallel computing ability [4]. It includes the process of creation of Spark context and session, distribution of the dataset on various nodes, and conversion of the dataset into DataFrame through which manipulation can be done over the data. Then, the preliminary optimal number of clusters is defined by the help of the Elbow method, which is eventually refined with the Silhouette analysis. This will compute the Silhouette score for several settings of clusters over several datasets, thus aiding us in arriving at the optimum number of clusters that provides better improvements in the compactness and separation of those clusters. Leverage the scalability and computing power of Spark with the rigorous evaluation Silhouette analysis affords us; this serves towards robust and resilient clustering solutions that can withstand many usual big data settings handling large-scale datasets.

The main contributions of this research work are summarized below

- Implementation of K-means clustering leveraging Apache Spark's distributed computing capabilities.
- Utilization of the Elbow method and Silhouette analysis to identify optimal cluster configurations validated across multiple datasets.
- Validation of findings across multiple datasets to ensure clustering accuracy and scalability.

The rest of the paper is organized as follows: Section 2 introduces the literature review work on the review study; Section 3 introduces related work using traditional techniques, Section 4 outlines the methodology approach; Section 5 discusses the results findings; Section 6 concludes the work.

2. LITERATURE REVIEW

The literature review conducted for the aforementioned study on Big Data clustering techniques encompasses a thorough examination of recent advancements in the field. It start from the evolution of clustering methodologies, tracing the trajectory from early suggested approaches to modern, sophisticated solutions tailored to the intricacies of Big Data analysis. Each reviewed solution is scrutinized for its main characteristics and drawbacks, providing valuable insights into the state-of-the-art techniques available for handling Big Data clustering challenges. By synthesizing findings from diverse sources, the literature review contributes to a comprehensive understanding of the landscape of clustering in Big Data analytics, enabling informed decision-making regarding the selection and implementation of clustering algorithms. Moreover, it serves as a foundation for identifying gaps in existing research and charting future directions for innovation in this critical domain.

Unsupervised Clustering

K-means serves as a foundational clustering framework, encompassing a family of distance functions that underpin various k-means algorithms. Its widespread adoption in clustering Big Data is attributed to its simplicity and rapid convergence (Jain et al. [8]). However, a notable drawback of k-means lies in the necessity of predefining the number of clusters, a factor that significantly influences the accuracy of the final classification. Moreover, k-means may not be suitable for scenarios where clusters exhibit non-convex distributions or vary in size (Jain et al., [8]). Recognizing these limitations, researchers have proposed several modifications to k-means, such as fuzzy k-means and k-means++ (Huang et al. [9]). Additionally, efforts have been directed towards optimizing k-means performance and scalability within the Spark framework through various research endeavors.

In Guo et al. [10], a weighted agglomerative hierarchical clustering algorithm is presented, specifically designed for analyzing residents' activities in China. The dataset relied on mobile phone connections with nearby stations, gathered and stored in Spark for analysis over a week's duration. Initially, the algorithm identified hot areas characterized by large populations, then proceeded to analyze pedestrian flow within each identified hotspot. The approach yielded meaningful insights at reduced cost and with greater accuracy compared to traditional investigation methods.

In Rotsnarani et al. [11], a comprehensive survey on the Hadoop framework for big data processing is presented. Various aspects of Hadoop's map-reduce capabilities are examined to address scalability and complexity challenges inherent in processing large datasets. Similarly, Rujal et al. [12] conducted a survey focusing on the utilization of the map-reduce model for implementing k-means clustering. The article delves into the technical intricacies of parallelizing k-means using Apache Hadoop, highlighting its viability for specific big data clustering applications, which has garnered significant attention from researchers. Conversely, Sood et al. [13] conducted a survey highlighting the primary challenges encountered in big data processing with Hadoop's map-reduce paradigm, emphasizing network latency as a prominent limitation.

In Manwal et al. [14], an extensive survey on big data and Hadoop architecture is provided. The paper categorizes existing Hadoop-based systems and evaluates their respective advantages and disadvantages. Furthermore, it elucidates various technologies such as Hbase, Hive, and Pig, employed in conjunction with the Hadoop Distributed File System (HDFS). Jiang et al. [15] conducted a survey focusing on large-scale data processing using Hadoop deployed over cloud infrastructure. The study elucidates the core components of the Hadoop platform and their functionalities, offering insights into its utilization for processing vast volumes of data in cloud environments.

Machine learning based methods

Kusuma et al. [16] introduced intelligent k-means, leveraging Spark, as a fully unsupervised learning algorithm capable of clustering data without prior knowledge of the number of clusters. Sarazin et al. [17] proposed a parallel implementation of biclustering using MapReduce over the Spark platform. Gao et al. [18] enhanced the selection process of k-means by combining Particle Swarm Optimization and Cuckoo-search within the Spark framework. Thakur et al. [19] presented a hybrid approach integrating k-means and decision trees for clustering and anomaly detection in big data. Initially, k-means is applied to the data to produce clusters, followed by the application of the decision tree algorithm to classify normal and anomaly instances within each cluster.

In Lighari et al. [20], the author has combined the rule-based and k-means algorithms to detect network anomalies by using Apache Spark. This study integrates a rule-based approach to derive known attacks and applies unsupervised learning using k-means for new, unknown attacks. The accuracy rate that was obtained was at 93%, following an evaluation that had been done on the KDD Cup dataset. Kamaruddin et al. [21] proposed parallelizing evolving methods with regard to clustering algorithms. The approach applied here, called EMC, is an online technique, processing every sample of data with just one pass and not requiring any iterative procedures. This makes the algorithm more efficient for the modern applications in real time where the data stream is dominated by high-dimensional data. The authors applied their algorithm on a large dataset of credit card frauds and demonstrated its supremacy over typical single EMC techniques.

Sarazin et al. [22] developed clustering algorithms suitable for MapReduce implementation on the Spark platform, with a specific emphasis on the widely used Self-organizing Map (SOM) clustering algorithm. Malondkar et al. [23] introduced Spark-GHSOM, an algorithm capable of scaling to handle large real-world datasets distributed across clusters. Additionally, their proposal includes a novel distance hierarchy method tailored for datasets with mixed attributes, resulting in the creation of a multi-level hierarchy of SOM layers.

Hosseini et al. [24] propose an algorithm for adaptive density estimation tailored for distributed big data applications, validated on prevalent datasets. This algorithm operates independently at each step, devoid of dependencies. Utilizing Bayesian Locality Sensitive Hashing (LSH), the input data is partitioned, with outliers filtered out through locality preservation, ensuring robustness. Moreover, clusters are rendered highly homogeneous through density definition based on Ordered Weighted Averaging distance. Corizzo et al. [25] introduce a scalable distributed density-based clustering method designed for conducting multi-regression tasks.

Fuzzy based methods

In Wu et al.'s study [26], a parallel implementation of fuzzy consensus clustering on the Spark platform was proposed for processing large-scale heterogeneous data. Win et al. [27] developed a crime pattern-discovery system based on fuzzy clustering under Spark, optimizing distance computations by employing the L2 norm instead of the Euclidean distance. In another work, Win et al. [28] utilized the fuzzy clustering method under Spark to identify potential criminal patterns in large-scale spatiotemporal datasets.

Liu et al. [29] presented a parallel algorithm for fuzzy-based image segmentation for agriculture-based big data applications. The algorithm uses the following steps: conversion of images into RGB format, allocation of such images into accessible nodes in the cloud, calculation of membership of pixel points for different cluster centroids, and then iteratively improving centroids until an optimal solution is obtained. The execution time on the Spark framework was much lower in comparison to the Hadoop-centric approach for evaluation. Bharill, Tiwari, and Malviya et al. [30] proposed a fuzzy c-Means algorithm that is a variant of Scalable Random Sampling with Iterative Optimization (SRSIO-FCM). Importantly, this

algorithm does not maintain the membership matrix, and this in turn contributes to lower execution times.

Scalable methods

Jin et al. [31] had developed a parallel algorithm for Single-linkage Hierarchical Clustering, which was modeled as a Minimum Spanning Tree problem. The algorithm was tested on two large datasets that showed different kinds of distributions, thus proving Spark's efficiency in parallelizing hierarchical clustering with remarkable scalability and performance. Solaimani et al. [32], in another publication, proposed a system to detect anomalies in multi-source VMware-based cloud data centres. This framework is constantly monitoring performance stream data coming from VMware, like CPU load and memory usage, for anomalies. Hassani et al. [33] provided an incremental hierarchical density-based stream clustering algorithm with a focus on stability in clusters.

Luo et al. [34] provided a parallelized approach of the DBSCAN algorithm; such parallelized version of it is known as S_DBSCAN and makes use of Spark. The described implementation consists of three major stages: initial partitioning of input data by random sampling, parallel local computations of DBSCAN that produce partial clusters, and then merging the partial clusters using centroid information. It well handles both integration and separation of resulting clusters of original data. Compare this with the serial version to demonstrate performance gains in processing large amounts of data on the Spark platform.

In a relevant study, Han et al. [35] proposed a scalable parallel execution of the DBSCAN algorithm under the Apache Spark framework, using a partitioning methodology that relies on kd-trees to reduce the search time. This approach is applied both on performance and scalability aspects because it produces well-balanced sub-domains among Spark executors. Baralis, Garza & Pastor (2018) extended the DBSCAN algorithm to Spark implementation. To get points representing the original data distribution and information about density, they added a preprocessing step before implementing the algorithm. This comes with the advantage of scaling up large data sets. Gong, Sinnott & Rimba [36] presented a real-time density-based clustering algorithm, more of a stream data analysis as an extension of DBSCAN. RT-DBSCAN uses spatiotemporal distance concepts for the clustering of spatio-temporal data. Implemented over Spark stream, this algorithm was tested using social media content. The Tables 1 provided offer a comprehensive compilation of research papers spanning various methodologies and techniques used in the domain of big data analytics.

Table 1. Comparative Analysis of Researchers Work

Ref.	Methodology	Category
4]	Exploring effective strategies for managing and utilizing Big Data	Big Data
5]	Comparative analysis of clustering techniques in Big Data environments	Big Data, Clustering
6]	Development of machine learning-based models for detecting intrusions in Big Data systems	Big Data, Machine Learning
7]	Investigating current trends in text-based clustering algorithms	Big Data, Clustering
8]	Revisiting clustering methodologies with insights beyond k-means	Clustering
9]	Implementing parallel DBSCAN for spatial data analysis on the Spark framework	Big Data, Clustering
	Deriving insights on urban behavior through mobile	Big Data

Ref.	Methodology	Category
10]	Big Data processing	
11]	A survey on leveraging Hadoop for extensive Big Data analysis	Big Data
12]	Applying k-means clustering techniques using MapReduce for data mining	Big Data, Clustering
13-15]	Reviewing advancements in Hadoop technologies, scheduling, and optimization techniques	Big Data, Hadoop
16-19]	Spark-powered clustering strategies, including hybrid models and anomaly detection	Big Data, Clustering
20-22]	Integrating rule-based frameworks with clustering for robust data analytics	Big Data, Clustering
23-31]	Innovative clustering methods like SOM, GHSOM, and fuzzy-based approaches for large datasets	Big Data, Clustering
32]	Real-time detection of anomalies in Big Data using Spark-based algorithms	Big Data, Anomaly Detection
33-37]	Advanced distributed clustering methods such as DBSCAN and fuzzy clustering using Spark	Big Data, Clusterin

3. RELATED WORK

Unsupervised Learning Algorithms:

A. K Means Clustering

The objective of the KMeans algorithm is to minimize the sum of squared distances between data points and their corresponding cluster centroids.

$$J = \sum_{i=1}^{k} \sum_{x \in C_i} ||x - \mu_i||^2$$

where J is the objective function, k is the number of clusters, Ci is the i^{th} cluster, μi is the centroid of cluster Ci, and $||x-\mu i||^2$ represents the squared Euclidean distance between data point x and centroid μi.

Step 1: Initialize k centroids randomly.
Step 2: Assign each data point to the nearest centroid.
Step 3: Update the centroids by taking the mean of all data points assigned to each centroid.
Step 4: Repeat steps 2 and 3 until convergence or for a specified number of iterations

B. DBSCAN Algorithm:

Step 1: Choose a point randomly that has not been visited and its neighborhood is determined by the radius eps.
Step 2: If the number of points in the neighborhood is below the threshold $minPts$, then mark the point as noise. Otherwise, mark the point as part of a new cluster.
Step 3: Expand the cluster by adding all reachable points to the cluster.
Step 4: Repeat steps 1-3 until all points are classified into clusters or noise.

C. K Medoids Algorithm

Similar to K Means, but uses medoids instead of centroids.

$$J = \sum_{i=1}^{k} \sum_{x \in C_i} ||x - m_i||^2$$

where mi is the medoid of cluster Ci
Step 1: Initialize k medoids randomly
Step 2: Assign each data point to the nearest medoid.
Step 3: Update the medoids by choosing the point that minimizes the total distance to other points in the cluster.
Step 4: Repeat steps 2 and 3 until convergence or for a specified number of iterations.

4. METHODOLOGY

A. System Architecture

Our proposed system starts by taking two independent datasets as input, namely the IRIS dataset and the MALL dataset. The process is initiated by setting a Spark session which provides the environment of distributed computing, thereby initiating easy acceleration and efficient processing of big data. After configuring the process, the system allows loading the datasets into it and subsequent processing stages can be performed. Then, these loaded datasets get converted into a vectorized format. This step is necessary as it allows the manifestation and representation of characteristics of each data point in a standardized numerical form, which can be well taken care of by machine learning algorithms. Vectorization encapsulates intrinsic properties within datasets where further analysis can then be performed. The data would then be transferred to strictly numerical format where consistency and compatibility could be maintained in further computation. This step standardizes the input, hence making it easy to apply mathematical and statistical methodologies. After the preprocessing of the data stage, the system employs the usage of the k-means clustering algorithm. This method classifies data points into clusters according to the similarity of features, hence presenting an exploratory analysis of the potential number of clusters within the datasets. The purpose of this step is to find natural patterns of grouping in the data that could be interesting or could serve as a basis to conduct further analysis. For the optimal number of clusters, two validation methods are employed by the algorithm: Elbow Method and Silhouette Analysis. The Elbow Method is a plot of WCSS against the number of clusters. It identifies that point from which the rate of decline of WCSS decreases considerably. This point is called the "elbow." In most cases, the optimum number of clusters lies at this point. Silhouette Analysis Compares the strength of association of data points inside the clusters with that inside the neighboring clusters. This measure ranges from -1 to 1. The better the clustering performance of that data point, the higher is the silhouette coefficient. Such a systematic approach helped the proposed system identify and recognize the optimal clustering of the IRIS and MALL datasets with the optimized strength of the Spark processing and the advanced methodologies in machine learning. This also besides enriching the knowledge of basic data structures also provides a scalable framework for clustering analysis over large datasets. Figure 1 depicts the progression of the working process.

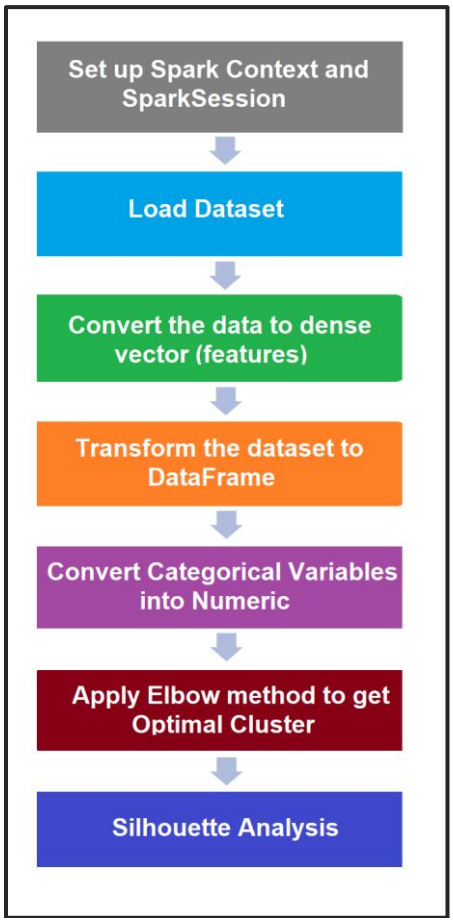

Figure 1. Working flow of K-Means Clustering Using Spark

B. Algorithms

The working of K-Means clustering using spark is shown below, where the csv file is used as input.

1. *K-means clustering Using Spark*

The k-means algorithm is a vector quantization algorithm originally from signal processing, widely used in cluster analysis in data mining. As the Expectation-Maximization algorithm shows, the k-means method works to solve the problem this way:

1. Assign some cluster centers
2. Repeated until converged
   - E-Step: assign points to the nearest center
   - M-step: set the cluster center to the mean

Given a set of observations $(x_1, x_2, \cdots, x_m)$. The objective function is

$$J = \sum_{i=1}^{m} \sum_{k=1}^{K} w_{ik} ||x_i - c_k||^2$$

where $w_{ik} = 1$if $x_i$is in cluster k; otherwise $w_{ik} = 0$and $c_k$is the centroid of x_i's cluster. Mathematically, k-means is a minimization problem with two parts: First, we minimize J w.r.t $w_{ik}$with $c_k$fixed; Then minimize Jw.r.t $c_k$with w_{ik}fixed. i.e.

E-step:

$$\frac{\partial J}{\partial w_{ik}} = \sum_{i=1}^{m}\sum_{k=1}^{K} ||x_i - c_k||^2$$

$$\Rightarrow w_{ik} = \begin{cases} 1, & \text{if } k = argmin_j ||x_i - c_j||^2 \\ 0, & \text{otherwise} \end{cases}$$

M-step:

$$\frac{\partial J}{\partial c_k} = 2\sum_{i=1} m w_{ik}(x_i - c_k) = 0 \Rightarrow c_k = \frac{\sum_{i=1}^{m} w_{ik} x_i}{\sum_{i=1}^{m} w_{ik}}$$

2. Proposed Algorithm of K-Means Clustering Step Using Spark:

Step 1. Set up Spark Context and SparkSession:

- Spark Context is the main entry point for Spark functionality. It represents the connection to the Spark cluster and can be used to create RDDs (Resilient Distributed Datasets).
- SparkSession is a unified entry point for reading data, configuring, and managing Spark functionality. It provides a way to interact with Spark SQL and DataFrames.

Step 2. Load dataset:

- In this step, we read the dataset into Spark. The dataset can be stored in various formats such as CSV, JSON, Parquet, etc. Spark provides APIs to load data from different sources like HDFS, local file system, or databases.

Step 3. Convert the data to dense vector (features):

- Machine learning algorithms in Spark expect data in a specific format, often as a vector of features. If your dataset contains multiple features, you need to combine them into a single dense vector. Spark provides `VectorAssembler` to do this efficiently.

Step 4. Transform the dataset to DataFrame:

- After converting the data into features, you'll typically want to work with it as a DataFrame, which is a distributed collection of data organized into named columns. DataFrames in Spark provide a higher-level abstraction and are easier to work with than RDDs.

Step 5. Deal with Categorical Variables:

- Categorical variables need to be converted into numerical values before applying machine learning algorithms. Spark provides various transformers like `StringIndexer` and `OneHotEncoder` to handle categorical variables.

Step 6. Elbow method to determine the optimal number of clusters for k-means clustering:

- The Elbow method is a heuristic used to determine the optimal number of clusters in a dataset for K-means clustering. It involves plotting the within-cluster sum of squares (WCSS) against the number of clusters and identifying the "elbow" point, where the rate of decrease in WCSS slows down. This point represents a good trade-off between the number of clusters and the within-cluster variance. The WCSS, also known as inertia, is calculated as follows:

$$\text{WCSS} = \sum_{k=1}^{K}\sum_{i=1}^{n} \|x_i^{(k)} - c_k\|^2$$

Where:

- K is the number of clusters.
- n is the number of data points in cluster k
- $x_i^{(k)}$ is a data point in cluster k

- C_k is the centroid of cluster k
- $|| x_i^{(k)} - C^{(k)}||^2$is the squared distance between a data point and the centroid of its cluster.

Step 7. Silhouette analysis:

- Silhouette analysis is another technique used to evaluate the quality of clusters created by clustering algorithms like K-means. It measures how similar an object is to its own cluster compared to other clusters. The silhouette score ranges from -1 to 1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. This analysis helps in understanding the separation distance between the resulting clusters. The formula for calculating the silhouette coefficient for a single data point (i) is:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

Where:

- $a(i)$ is the average distance from the i th data point to all other points in the same cluster (intra-cluster distance).
- $b(i)$ is the minimum average distance from the i th data point to points in a different cluster, minimized over clusters (nearest-cluster distance).

5. RESULT AND DISCUSSION

A. Experimental Setup

Python is used as development technology. Python, in particular, boasts a rich ecosystem of packages such as NumPy, Keras, Pandas, TensorFlow, and SciPy, which greatly facilitate scientific and numerical computations. Anaconda, an open-source distribution, simplifies package management and offers a suite of tools including Jupyter, Spyder, and RStudio for streamlined programming workflows.

Apache Hadoop and Spark emerge as prominent platforms for executing MapReduce jobs in big data analytics. PySpark, a tool within the Spark ecosystem, enables seamless integration with Resilient Distributed Databases (RDDs) in Python through the Py4j library. Numerous online tools are available to assist researchers in implementing Machine Learning Techniques (MLT) for Big Data Analytics, streamlining the development and deployment of their projects.

B. Dataset Description

Dataset 1: (Mall Customer) – The dataset is downloaded from https://www.kaggle.com/code/heeraldedhia/kmeans-clustering-for-customer-data/input . It contains basic information (ID, age, gender, income, spending score) about the customers. Following figure 2 shows the dataset values.

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

Figure 2. Mall Customer Dataset Feature and Values

Dataset 2: (IRIS) – The dataset is taken from *https://www.kaggle.com/datasets/uciml/iris* t includes three iris species with 50 samples each as well as some properties about each flower. One flower species is linearly separable from the other two, but the other two are not linearly

separable from each other. The columns in this dataset are: Id, SepalLengthCm, SepalWidthCm, PetalLengthCm, PetalWidthCm, Species.

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

Figure 3. IRIS Dataset Feature and Values

C. Result Analysis

In our study, we conducted K-means clustering using the Spark framework on two distinct datasets: the IRIS dataset and the Mall dataset. Leveraging the Silhouette and Elbow methods, we determined the optimal number of clusters for each dataset, ensuring robust clustering solutions.

I. MALL Dataset

The figure 4 presented depict the optimized clustering results obtained through Silhouette analysis, showcasing the compactness and separation of clusters. Additionally, another figure 5 illustrates the predicted clusters, while figure 6 provide a visual representation of the clustering outcomes. These visualizations offer insights into the distinct groupings within the datasets and aid in understanding the underlying patterns and structures present.

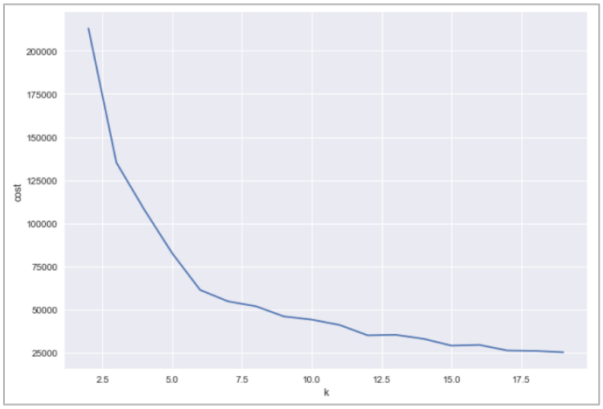

Figure 4. Optimised Cluster Choice for Mall Dataset using Silhouette (K=5)

Id	Age	AnnIncome	SpendScore	prediction
1	19.0	15.0	39.0	1
2	21.0	15.0	81.0	1
3	20.0	16.0	6.0	0
4	23.0	16.0	77.0	1
5	31.0	17.0	40.0	1

Figure 5. Result Prediction for Optimum Cluster Value (k=5) on Mall Dataset

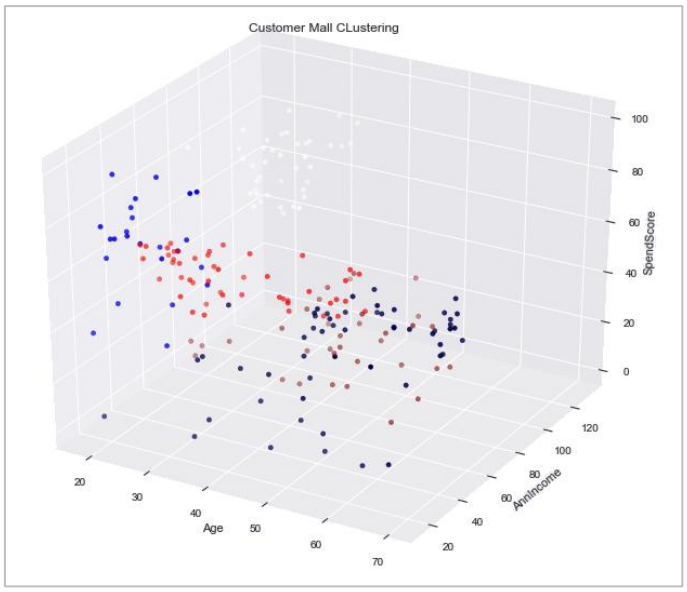

Figure 6. Cluster Visualisation (Mall Dataset)

II. IRIS Dataset

The figure 7 (A) presented depict the optimized clustering results obtained through Elbow method. Additionally, another Figure 7(B) illustrates the optimized clustering results obtained through Silhouette analysis, while Figure 8 provide a visual representation of the clustering outcomes.

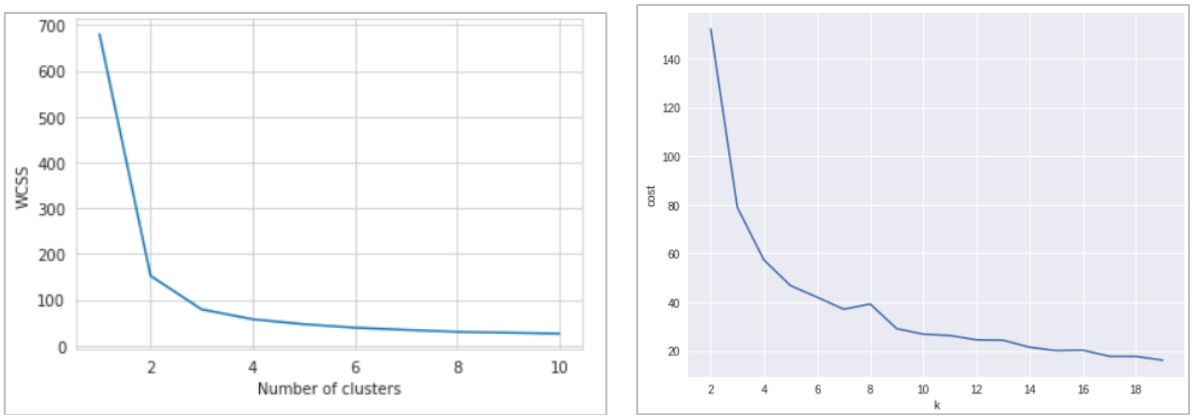

Figure 7(A). Optimised Cluster Choice for IRIS Dataset Using Elbow Method (K=3) and Figure 7(B). Optimised Cluster Choice for IRIS Dataset Using Silhouette (K=3)

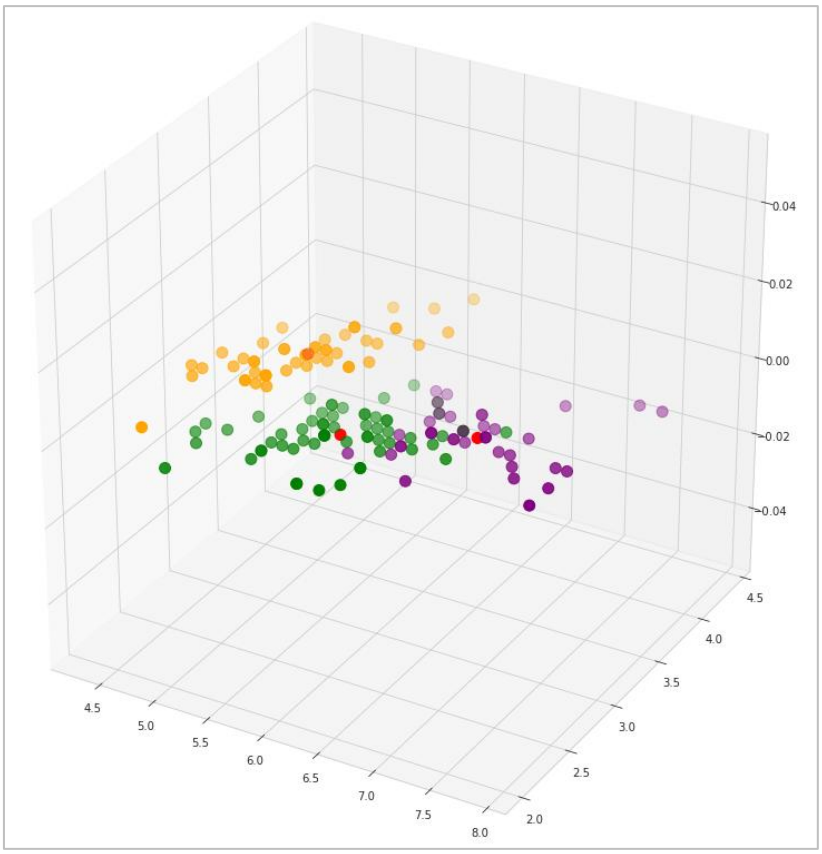

Figure 8. Cluster Visualisation (IRIS Dataset)

Overall, our analysis demonstrates the efficacy of Spark-based K-means clustering in partitioning datasets and provides valuable insights for further exploration and interpretation of the data.

6. CONCLUSION

This paper provides an overview of the challenges posed by the ever-growing volume and complexity of big data and the limitations of traditional learning algorithms in handling such data. Our study emphasizes the importance of leveraging Spark-enabled K-means clustering for efficient processing of large-scale datasets in big data analytics. By integrating Silhouette analysis and elbow method, we identify optimal cluster configurations, thereby enhancing clustering accuracy. This approach is validated across multiple datasets, highlighting its effectiveness for diverse data collections. Our work underscores the significance of employing parallel processing and leveraging frameworks like Spark to address the complexities and scale of big data analytics. These findings contribute to the ongoing development of analytical methods capable of effectively handling the challenges posed by big data.

7. DISCUSSION AND FUTURE WORK

Hybrid machine learning algorithms emerge as a promising avenue for big data analytics, offering the potential for improved results. Researchers have successfully implemented techniques like Deep Learning, Representation Learning, Online Learning, and Incremental Learning in the context of big data analytics. However, further exploration and study are warranted to fully harness their capabilities and potential in addressing the challenges posed by big data.

References

[1]. M. Assuncao, R. Calheiros, S. Bianchi, M. Netto and R. Buyya, "Big Data Computing and Clouds: Trends and Future Directions," Journal of Parallel and Distributed Computing, Elsevier, vol 79-80, pp. 3-15, 2015.

[2]. G. E. Hinton, S. Osindero and Y.W. Teh, "A Fast Learning Algorithm for Deep Belief Nets," Neural Computation, pp. 1527-1554, 2006.

[3]. Gul, M., Rehman, M. Big data: an optimized approach for cluster initialization. J Big Data 10, 120 (2023). https://doi.org/10.1186/s40537-023-00798-1

[4]. Liu, Haibo & Bai, Yongbin & Chen, Zhenhao & Zhang, Zhenfeng. (2024). Big data clustering method based on parallel K-means. 893-897. 10.1109/ICPECA60615.2024.10470970.

[5]. Dave & Gianey (2016) Dave M, Gianey H. Different clustering algorithms for Big Data analytics: a review. International conference system modeling & advancement in research trends (SMART), Moradabad; 2016. pp. 328–333.

[6]. Othman et al. (2018) Othman SM, Ba-Alwi FM, Alsohybe NT, Al-Hashida AY. Intrusion detection model using machine learning algorithm on Big Data environment. Journal of Big Data. 2018;5:34. doi: 10.1186/s40537-018-0145-4.

[7]. Fasheng & Xiong (2011) Fasheng L, Xiong L. Survey on text clustering algorithm - Research present situation of text clustering algorithm. 2011 IEEE 2nd international conference on software engineering and service science, Beijing; Piscataway. 2011. pp. 196–199.

[8]. Jain (2010) Jain AK. Data clustering: 50 years beyond k-means. Pattern Recognition Letters. 2010;31(8):651–666. doi: 10.1016/j.patrec.2009.09.011.

[9]. Huang et al. (2017) Huang F, Zhu Q, Zhou J, Tao J, Zhou X, Jin D, Tan X, Wang L. Research on the parallelization of the DBSCAN clustering algorithm for spatial data mining based on the spark platform. Remote Sensing. 2017;9(12):1301. doi: 10.3390/rs9121301.

[10]. Guo, Zhang & Zhang (2016) Guo Y, Zhang J, Zhang Y. An algorithm for analyzing the city residents' activity information through mobile big data mining. IEEE Trustcom/BigDataSE/ISPA, Tianjin; Piscataway. 2016. pp. 2133–2138.

[11]. Rotsnarani & Mrutyunjaya (2015) Rotsnarani S, Mrutyunjaya P. Big data analysis using Hadoop: a survey. International Journal of Advanced Research in Computer Science and Software Engineering. 2015;5(7):1153–1157.

[12]. Rujal & Dabhi (2016) Rujal B, Dabhi D. Extensive survey on k-means clustering using mapreduce in datamining. Conference: international conference on electronics and communication systems (ICECS) At: Coimbatore, Tamilnadu, India.2016.

[13]. Sood & Singh (2019) Sood S, Singh R. A survey of performance improvement techniques for hadoop. International Journal of Applied Engineering Research. 2019;10(55):2481–2486.

[14]. Manwal & Gupta (2017) Manwal M, Gupta A. Big data and hadoop—a technological survey. International conference on emerging trends in computing and communication technologies (ICETCCT), Dehradun; 2017. pp. 1–6.

[15]. Jiang et al. (2010) Jiang D, Ooi B, Shi L, Wu S. Big data processing using hadoop: survey on scheduling. Proceedings of the VLDB Endowment. 2010;3(10):272–277.

[16]. Kusuma et al. (2016) Kusuma I, Ma'sum MA, Habibie N, Jatmiko W, Suhartanto H. Design of intelligent k-means based on spark for big data clustering. International workshop on big data and information security (IWBIS), Jakarta; 2016. pp. 89–96.

[17]. Sarazin, Lebbah & Azzag (2014) Sarazin T, Lebbah M, Azzag H. Biclustering using spark-mapreduce. IEEE international conference on big data (Big Data), Washington, DC; Piscataway. 2014. pp. 58–60.

[18]. Gao & Zhang (2017) Gao ZQ, Zhang LJ. DPHKMS: an efficient hybrid clustering preserving differential privacy in spark. International conference on emerging internetworking, data & web technologies.2017.

[19]. Thakur & Dharavath (2018) Thakur S, Dharavath R. Information and decision sciences. vol. 701. Springer; Singapore: 2018. KMDT: a hybrid cluster approach for anomaly detection using big data.

[20]. Lighari & Hussain (2017) Lighari SN, Hussain DMA. Hybrid model of rule based and clustering analysis for big data security. First international conference on latest trends in electrical engineering and computing technologies (INTELLECT), Karachi; 2017. pp. 1–5.

[21]. Kamaruddin, Ravi & Mayank () Kamaruddin S, Ravi V, Mayank P. Parallel evolving clustering method for big data analytics using apache spark: applications to banking and physics. In: Reddy P, Sureka A, Chakravarthy S, Bhalla S, editors. Big data analytics. BDA 2017. vol. 10721. Springer.

[22]. Sarazin, Azzag & Lebbah (2014) Sarazin T, Azzag H, Lebbah M. SOM clustering using spark-mapReduce. 2014 IEEE international parallel & distributed processing symposium workshops, Phoenix, AZ; Piscataway. 2014. pp. 1727–1734.

[23]. Malondkar et al. (2019) Malondkar A, Corizzo R, Kiringa I, Ceci M, Japkowicz N. Spark-GHSOM: growing hierarchical self-organizing map for large scale mixed attribute datasets. Information Sciences. 2019;496:572–591. doi: 10.1016/j.ins.2018.12.007.

[24]. Hosseini & Kiani (2018) Hosseini B, Kiani K. A robust distributed big data clustering-based on adaptive density partitioning using apache spark. Symmetry. 2018;10:342. doi: 10.3390/sym10080342.

[25]. Corizzo et al. (2019) Corizzo R, Pio G, Ceci M, Malerba D. DENCAST: distributed density-based clustering for multi-target regression. Journal of Big Data. 2019;6(1):43. doi: 10.1186/s40537-019-0207-2.

[26]. Wu et al. (2017) Wu J, Wu Z, Cao J, Liu H, Chen G, Zhang Y. Fuzzy consensus clustering with applications on big data. IEEE Transactions on Fuzzy Systems. 2017;25(6):1430–1445. doi: 10.1109/TFUZZ.2017.2742463.

[27]. Win et al. (2019a) Win KN, Chen J, Chen Y, Fournier-Viger P. PCPD: a parallel crime pattern discovery system for large-scale spatiotemporal data based on fuzzy clustering. International Journal of Fuzzy Systems. 2019a;21:1961–1974. doi: 10.1007/s40815-019-00673-3.

[28]. Win et al. (2019b) Win KN, Chen J, Xiao G, Chen Y, Viger PF. A parallel crime activity clustering algorithm based on apache spark cloud computing platform. IEEE 21st international conference on high performance computing and communications; Zhangjiajie, China; 2019b. pp. 68–74.

[29]. Liu et al. (2019) Liu B, He S, He D, Zhang Y, Guizani M. A spark-based parallel fuzzy c -means segmentation algorithm for agricultural image big data. IEEE Access. 2019;7:42169–42180. doi: 10.1109/ACCESS.2019.2907573.

[30]. Bharill, Tiwari & Malviya () Bharill N, Tiwari A, Malviya A. Fuzzy based scalable clustering algorithms for handling big data using apache spark. IEEE Transactions on Big Data. 2016:1–1. doi: 10.1109/TBDATA.2016.2622288.

[31]. Jin et al. (2015) Jin C, Liu R, Chen Z, Hendrix W, Agrawal A, Choudhary A. A scalable hierarchical clustering algorithm using spark. IEEE first international conference on big data computing service and applications, Redwood City, CA; Piscataway. 2015. pp. 418–426.

[32]. Solaimani et al. () Solaimani M, Iftekhar M, Khan L, Thuraisingham B, Ingram JB. Spark-based anomaly detection over multi-source VMware performance data in real-time. 2014 IEEE symposium on computational intelligence in cyber security (CICS);

[33]. Hassani et al. (2016) Hassani M, Spaus P, Cuzzocrea A, Seidl T. I-hastream: density-based hierarchical clustering of big data streams and its application to big graph analytics tools. 16th IEEE/ACM international symposium on cluster, cloud and grid computing (CCGrid); Piscataway. 2016. pp. 656–665.

[34]. Luo et al. (2016) Luo G, Luo X, Gooch TF, Tian L, Qin K. A parallel DBSCAN algorithm based on spark. IEEE international conferences on big data and cloud computing (BDCloud), social computing and networking (SocialCom), sustainable computing and communications (SustainCom) (BDCloud-SocialCom-SustainCom), Atlanta, GA; Piscataway. 2016. pp. 548–553.

[35]. Han et al. (2018b) Han D, Agrawal A, Liao W, Choudhary A. Parallel DBSCAN algorithm using a data partitioning strategy with spark implementation. IEEE international conference on big data (Big Data), Seattle, WA, USA; Piscataway. 2018b. pp. 305–312.

[36]. Gong, Sinnott & Rimba Gong Y, Sinnott RO, Rimba P. RT-DBSCAN: real-time parallel clustering of spatio-temporal data using spark-streaming. In: Shi Y, et al., editors. Computational science—ICCS 2018. ICCS 2018. vol. 10860.

[37]. K Rajendra Prasad, Moulana Mohammed, LV Narasimha Prasad, and Dinesh Kumar An-guraj. An efficient sampling-based visualization technique for big data clustering with crisp partitions. Distributed and Parallel Databases, 39(3):813–832, 2021.