Performance Optimization of K-Means Clustering using multiple k-Values:A Hands-On

SHIVARAJ BG[1], AMRUTHA[2], SHWETHA KAMATH[3]
[1, 2, 3] *Department of Computer Science and Engineering, Mangalore Institute of Technology & Engineering*

Abstract- ***Optimizing the performance of K-means clustering involves several techniques and strategies that can help speed up the computation and improve the clustering quality. Distortion and inertia are key metrics used to evaluate the quality, assess the clustering performance, and determine the optimal number of clusters of K-means clustering. Using the Elbow Method, we can plot Distortion and Inertia metrics against different values of 𝑘 to determine the optimal number of clusters. This approach helps achieve better clustering results by ensuring that data points are grouped most meaningfully.***

Indexed Terms- ***K-Means Clustering, Distortion, Inertia, Elbow Method.***

I. INTRODUCTION

K-means clustering widely used unsupervised machine learning algorithm for grouping data points into clusters based on their similarity. To train the model, there is no need for labeled data. The data patterns are identified and grouped into data points based on their inbuilt characteristics and partition a dataset into a pre-defined number of clusters denoted by ‘k’. K-means clustering uses the concept of distance and aims to minimize the distance between data points within the same cluster while maximizing the distance between clusters. Using distortion and inertia, effectively evaluate the quality of K-means clustering. The Elbow Method provides a visual representation and instinctive way to determine the optimal number of clusters, ensuring superior clustering results and more significant data distribution which helps Data Exploration, Data Segmentation, Anomaly Detection, and Image Segmentation.

II. LITERATURE REVIEW

[1] K-means Clustering Algorithms: A Comprehensive Review, Variants Analysis, and Advances in the Era of Big Data": This paper discusses the original K-means algorithm and its variants, addressing improvements like k-means++, scalable approaches, and hybrid models with other algorithms. It also covers the algorithm's limitations and solutions proposed in recent studies to enhance clustering performance and efficiency.

[2] "The K-means Algorithm: A Comprehensive Survey and Performance Evaluation": This survey highlights the challenges of random centroid initialization and the requirement to predefine the number of clusters. It evaluates several variants of the K-means algorithm, emphasizing experimental analyses on diverse datasets to compare their performance and effectiveness.

[3] "Optimizing K-means for Big Data: A Comparative Study": The focus is on optimizing K-means for large datasets through parallel processing, distributed computing, and the use of advanced initialization techniques. The study also explores the integration of K-means with big data frameworks like Hadoop and Spark to improve scalability and processing speed.

[4] "Boosting K-means Clustering with Symbiotic Organisms Search for Better Performance": This paper explores the hybridization of K-means with nature-inspired metaheuristic algorithms, such as the Symbiotic Organisms Search (SOS), to overcome issues like local optima and slow convergence. The results show significant improvements in clustering quality and computational efficiency.

[5] "A Comprehensive Survey of Clustering Algorithms: State-of-the-Art Analysis and Emerging Trends": It provides a broad overview of clustering techniques, with a detailed comparison of K-means

against other methods. The survey highlights the algorithm's adaptability to various domains, including cybersecurity and healthcare, and discusses future directions for research in clustering methodologies.

III. METHODOLOGY

Working procedure of K-Means Clustering

Step 1: Initialization- Randomly select 'k' data points as initial cluster centroids.

Step 2: Assignment- Assign each data point to the nearest centroid based on its distance.

Step 3: Update- Recalculate the centroid of each cluster by averaging the data points assigned to that cluster.

Step 4: Repeat- Repeat the assignment and update steps until the cluster assignments stabilize, indicating convergence.

Step 5: Distortion = 1/n * Σ(distance(point, centroid)^2).

Step 6: Inertia = Σ(distance(point, centroid)^2).

Step 7: Python code for implementing K-Means Clustering.

7.1 Importing the required libraries

```
from sklearn.cluster import KMeans
from sklearn import metrics
from scipy.spatial.distance import cdist
import numpy as np
import matplotlib.pyplot as plt
#Creating and Visualizing the data
# Creating the data
x1 = np.array([3, 1, 1, 2, 1, 6, 6, 6, 5, 6,\
                    7, 8, 9, 8, 9, 9, 8, 4, 4, 5, 4])
x2 = np.array([5, 4, 5, 6, 5, 8, 6, 7, 6, 7, \
                        1, 2, 1, 2, 3, 2, 3, 9, 10, 9, 10])
X = np.array(list(zip(x1, x2))).reshape(len(x1), 2)
# Visualizing the data
plt.plot()
plt.xlim([0, 10])
plt.ylim([0, 10])
plt.title('Dataset')
plt.scatter(x1, x2)
plt.show()
```

Output: Refer Figure 1

7.2 Building the clustering model and calculating the values of the Distortion and Inertia

```
distortions = [ ]
inertias = [ ]
mapping1 = {}
mapping2 = {}
K = range(1, 10)
for k in K:
    # Building and fitting the model
    kmeanModel = KMeans(n_clusters=k).fit(X)
    kmeanModel.fit(X)
    distortions.append(sum(np.min(cdist(X, kmeanModel.cluster_centers_,'euclidean'), axis=1)) / X.shape[0])
    inertias.append(kmeanModel.inertia_)
    mapping1[k] = sum(np.min(cdist(X, kmeanModel.cluster_centers_,'euclidean'), axis=1)) / X.shape[0]
    mapping2[k] = kmeanModel.inertia_
for key, val in mapping1.items():
    print(f'{key} : {val}')
```

7.3 The Elbow Method using Distortion

```
plt.plot(K, distortions, 'bx-')
plt.xlabel('Values of K')
plt.ylabel('Distortion')
plt.title('The Elbow Method using Distortion')
plt.show()
```

Output: Refer Figure 2

7.4 The Elbow Method using Inertia

```
plt.plot(K, inertias, 'bx-')
plt.xlabel('Values of K')
plt.ylabel('Inertia')
plt.title('The Elbow Method using Inertia')
plt.show()
```

Output: Refer Figure 3

7.5 Code for K for 1,2,3, 4 and Elbow Method

```
import matplotlib.pyplot as plt
# Create a range of values for k
k_range = range(1, 5)
# Initialize an empty list to store the inertia values for each k
inertia_values = []
# Fit and plot the data for each k value
for k in k_range:
    kmeans = KMeans(n_clusters=k, \
    init='k-means++', random_state=42)
    y_kmeans = kmeans.fit_predict(X)
    inertia_values.append(kmeans.inertia_)
    plt.scatter(X[:, 0], X[:, 1], c=y_kmeans)
    plt.scatter(kmeans.cluster_centers_[:, 0],\
```

```
        kmeans.cluster_centers_[:, 1], \
                                    s=100, c='red')
        plt.title('K-means                clustering
(k={})'.format(k))
        plt.xlabel('Feature 1')
        plt.ylabel('Feature 2')
        plt.show()
# Plot the inertia values for each k
plt.plot(k_range, inertia_values, 'bo-')
plt.title('Elbow Method')
plt.xlabel('Number of clusters (k)')
plt.ylabel('Inertia')
plt.show()
```

Output: Refer Figures 4 – 8

IV. RESULTS AND CONCLUSION

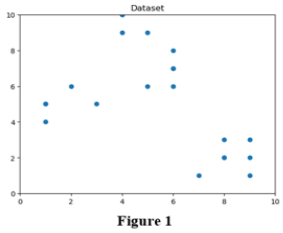

Figure 1

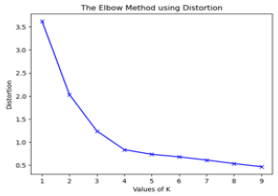

Figure 2

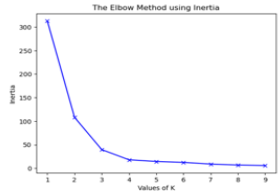

Figure 3

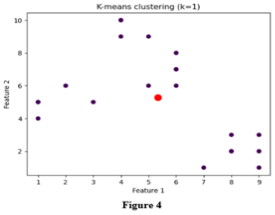

Figure 4

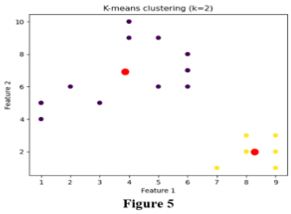

Figure 5

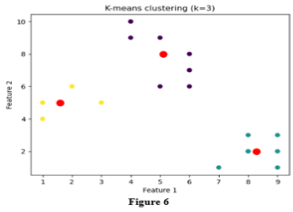

Figure 6

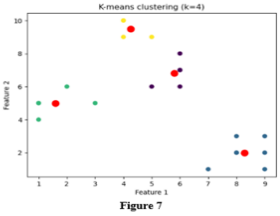

Figure 7

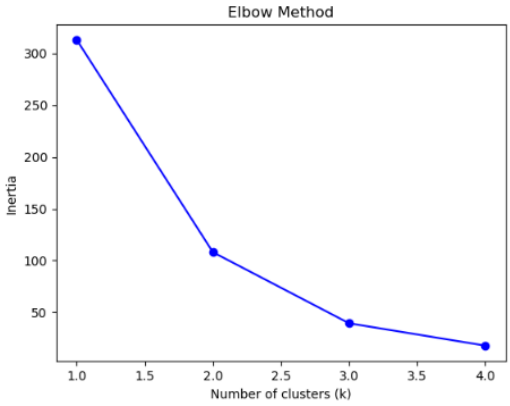

Figure 8

CONCLUSION

Conclusion for The Elbow Method

Varying K Values: Run K-means clustering with different values of 'k', ranging from a small number to a larger number.

Calculate Within-Cluster Sum of Squares (WCSS): For each 'k', calculate the sum of squared distances between each data point and its corresponding cluster centroid.

Plot WCSS vs. K: Plot the WCSS values against the corresponding 'k' values. The resulting curve typically exhibits an "elbow" shape.

Identify the Optimal K: The optimal 'k' value is determined by the "elbow" point of the curve, where the decrease in WCSS starts to level off.

ACKNOWLEDGMENT

We would like to acknowledge the contributions of the numerous researchers and developers who have advanced the field of clustering algorithms, particularly K-means. Their extensive studies and innovations have provided valuable insights into the optimization and application of K-means clustering across various domains. Additionally, we appreciate the availability of open-source libraries and datasets that facilitated the practical implementation and experimentation described in this work.

REFERENCES

[1] Abiodun M. Ikotun,Absalom E. Ezugwu, Laith Abualigah, Belal Abuhaija, Jia Heming: K-means Clustering Algorithms: A

Comprehensive Review, Variants Analysis, and Advances in the Era of Big Data".

[2] Mohiuddin Ahmed, Raihan Seraj, Syed Mohammed Shamsul Islam: "The K-means Algorithm: A Comprehensive Survey and Performance Evaluation".

[3] Ravil Mussabayev, Rustam Mussabayev: "Comparative Analysis of Optimization Strategies for K-means Clustering in Big Data Contexts: A Review".

[4] Abiodun M. Ikotun, Absalom E. Ezugwu:"Boosting K-means Clustering with Symbiotic Organisms Search for Better Performance.