Assisted Requirements Selection by Clustering using an Analytical Hierarchical Process

Shehzadi Nazeeha Saleem[1], Linda Mohaisen[2]
Department of Computer Science and Software Engineering,
National University of Sciences and Technology, Islamabad, Pakistan[1]
Department of Information Technology, Faculty of Computing and Information Technology,
King Abdulaziz University, Jeddah, Saudi Arabia[2]
Department of Computer Science, Cardiff Metropolitan University, Cardiff CF5 2YB, UK[2]

***Abstract*—This research investigates the fusion of the Analytic Hierarchy Process (AHP) with clustering techniques to enhance project outcomes. Two quantitative datasets comprising 20 and 100 software requirements are analyzed. A novel AHP dataset is developed to impartially evaluate clustering strategies. Five clustering algorithms (K-means, Hierarchical, PAM, GMM, BIRCH) are employed, providing diverse analytical tools. Cluster quality and coherence are assessed using evaluation criteria including the Dunn Index, Silhouette Index, and Calinski Harabaz Index. The MoSCoW technique organizes requirements into clusters, prioritizing critical requirements. This strategy combines strategic prioritization with quantitative analysis, facilitating objective evaluation of clustering results and resource allocation based on requirement priority. The study demonstrates how clustering can prioritize software requirements and integrate advanced data analysis into project management, showcasing the transformative potential of converging AHP with clustering in software engineering.**

Keywords*—Requirements prioritization; next release plan; software product planning; decision support; MoSCoW; AHP; k-Means; GMM; BIRCH; PAM; hierarchical; clustering; clusters evaluation

I. Introduction

Software engineering is built on several pillars and involves more than just programming. It contains every piece of supporting information, design principle, or idea required to make these programmes function as intended. Software requirements prioritisation (SRP) is one of the design principles that enable software that is being considered for development to function as intended [1].

A subfield of requirements engineering called requirements prioritisation assists in selecting requirements based on the interests of stakeholders. Giving each requirement a priority to decide the order in which they should be implemented is a step in the software engineering process. A requirement engineering decision process is used to decide which features or requirements will be developed in the upcoming release while considering technical, resource, risk, and budget constraints [2]. Choosing the order in which requirements should be addressed is a crucial step in the software development process. This process aids in managing the priority and urgency of software requirements while considering stakeholders' interest, cost, resource, and time issues. Numerous academics have provided definitions for the ranking of software demands in order of importance. Software requirement prioritisation is a process that determines the order in which needs will be implemented [3]. The process of selecting the best set of requirements from several conflicting and competing expectations gathered from various stakeholders participating in a software development project, according to Karlsson and Ryan [4].

The success or failure of a project is largely dependent on the software requirements specification in general and the prioritisation of software requirements in particular. Almost 80% of software projects fail to achieve the Standish Group's definitions of success based on time, cost, and scope criteria each year [5]. The failure is often due to shifting requirements, as requirements are often documented and rarely changed. This suggests that software projects fail due to their inability to evolve efficiently to match shifting requirements or accommodate new ones. This highlights the importance of release management and the need for proper decision-making about the functionality of a software product's release. A well-selected release will minimize problems with shifting requirements in future releases.

As a remedy to this issue, many requirements prioritisation techniques have been put forth. These techniques aim to reduce the length and cost of software development projects by supporting developers in identifying the most important and urgent requirements. Each method has limitations and makes both explicit and implicit assumptions about the project context during requirements prioritisation [6]. These presumptions must be considered while experimentally assessing a requirement prioritisation approach for usefulness, utility, application, or effectiveness.

One technique for ranking software requirements is to use clustering techniques. Similar observations, data points, or feature vectors can be clustered together based on shared characteristics using the clustering technique [7]. Clustering algorithms are used in the prioritising process to group and categorise requirements based on similarity or relatedness. This enables effective requirements prioritisation based on the characteristics of each cluster and the discovery of patterns and relationships between them. Clustering algorithms can assist in managing the complexity of prioritising various requirements by organising requirements into meaningful clusters that can then be prioritised more successfully.

This study thoroughly explores an innovative and promising method for requirement prioritisation that combines the Analytic Hierarchy Process (AHP) and clustering techniques. With the use of the data mining approach known as clustering, it may be possible to group together requirements that are similar, making it easier to handle them and improving the decision-making process. AHP, on the other hand, is a structured method for making decisions based on several factors and enables the creation of priorities based on both qualitative and quantitative judgments.

As we seek to assess the accuracy of quantitative records, it is crucial to assign requirements the proper level of importance to determine the core set of requirements. To do this, the MoSCoW technique, a tried-and-true framework for prioritising requirements, is used that divides each into Must-haves, Should-haves, Could-haves, and Won't-haves categories based on how important and consequential they are. A robust evaluation framework is also developed using metrics like the Dunn Index, Silhouette Index, and Calinski Harabaz Index. These metrics provide quantitative insights into the quality and cohesion of clusters, aiding decision-making processes.

Let's suppose a software development team is tasked with prioritizing features for an e-commerce platform using clustering techniques. They assign priorities within each cluster based on business impact and technical complexity. For example, they prioritize product search functionality (Cluster A) and payment processing (Cluster B) based on their significance for user experience and revenue generation. This approach streamlines decision-making, ensuring high-priority features align with business goals and user needs, ultimately optimizing the software development process.

Our overarching objective in this research is to evaluate the results of combining clustering methods with the Analytic Hierarchy Process (AHP). Our view of this integration's potential impact will be greatly influenced by the outcomes of this integration, which are expected to provide a distinctive perspective on requirement prioritisation and project management. In keeping with this goal, we have developed two key research questions that will direct our empirical studies and provide the information required to make well-informed decisions.

RQ1: Is a semi-automated approach to SRP processes possible with the incorporation of clustering techniques?

RQ2: Does the fusion of AHP and clustering generate better results?

The remainder of the paper is structured as follows: It commences with the state of the art for clustering algorithms and prioritisation techniques in Section II. Following this, Section III gives an overview of established techniques for clustering and requirements prioritisation. In Section IV, we elaborate on the methodology proposed for clustering requirements using AHP including how to determine the number of clusters, evaluate clusters, and associate MoSCoW categories with them. Section V presents and analyzes the results of an effectiveness study conducted on two different datasets. Section VI is dedicated to addressing the effectiveness of the proposed method. Lastly, Section VII presents the Results. Section VIII encapsulates the conclusions drawn from the research.

II. LITRUTURE REVIEW

Table I extensively evaluates several works on algorithms for clustering and requirements prioritisation. Notably, a wide range of techniques were investigated within the state of the art, including Binary Search Tree, Analytic Network Process, Spanning Tree, Numerical Analysis, Bubble Sort, MoSCoW, and Analytical Hierarchical Process. Remarkably, the Analytical Hierarchical Process (AHP) was the method of choice among researchers due to its constant production of superior results. The section also discusses several clustering techniques, such as K-Means, Partition Around Medoids, BIRCH, Agglomerative Hierarchical Clustering, and Gaussian Mixture Model (GMM). This review of the literature provides an overview of the field and paves the way for the creation of an original and useful framework, laying the groundwork for succeeding research phases.

TABLE I. LITERATURE REVIEW

Year	*Title*	*Techniques Used*	*Results*	*Ref.*
2015	Applying the analytical hierarchy process to system quality requirements prioritisation	AHP	The AHP technique effectively removes discrepancies between stakeholders' interests and the business goals.	[8]
2015	Comparison of Requirement Prioritisation Techniques to Find the Best Prioritisation Technique	binary search tree, AHP, hierarchy AHP, spanning tree matrix, priority group/Numerical Analysis, bubble sort, MoSoW, simple ranking, and Planning Game		[9]
2016	An Evaluation of Requirement Prioritisation Techniques with ANP	ANP, binary search tree, AHP, hierarchy AHP, spanning tree matrix, priority group and bubble sort	AHP is the best requirements prioritisation technique amongst all the requirements prioritisation techniques	[10]
2016	An approach to the estimation of the degree of customization for ERP projects using prioritised requirements	Framework using AHP	AHP framework gave better results	[11]
2017	Fuzzy_MoSCoW: A fuzzy based MoSCoW method for the prioritisation of software requirements	Fuzzy MoSCoW	ANP is the best technique among the seven techniques,	[12]

Year	Title	Techniques Used	Results	Ref.
			though it consumes time	
2020	A Novel Approach for Software Requirement Prioritisation	MAHP, a combination of AHP and MoSCoW		[13]
2020	Prioritisation of Software Functional Requirements from Developers' Perspective	Spanning Tree and AHP	AHP framework gave better results	[14]
2022	E-AHP: An Enhanced Analytical Hierarchy Process Algorithm for Prioritising Large Software Requirements Numbers	Enhanced AHP	E-AHP gives better results for large projects	[15]
2015	Efficient agglomerative hierarchical clustering	Efficient agglomerative hierarchical clustering	Experimental results show consistent performance across various settings, proving efficient AHP to be reliable.	[16]
2016	A hierarchical clustering method for multivariate geostatistical data	Aglomerative hierarchical clustering	Proposed clustering method yields satisfactory results compared to other geostatistical methods.	[17]
2017	Milling tool wear state recognition based on partitioning around medoids (PAM) clustering	PAM	PAM outperforms k-means and fuzzy c-means in Ti-6Al-4V alloy end milling experiments.	[18]
2017	Malware family identification with BIRCH clustering	BIRCH	BIRCH excels in malware family identification with high accuracy and low clustering time.	[19]
2020	Unsupervised K-Means Clustering Algorithm	Unsupervised K-Means	The U-k-means algorithm is robust to data structure and performs better than existing algorithms.	[20]
2020	Applications of Clustering Techniques in Data Mining: A Comparative Study	K-Means, Hierarchical Clustering, DB Scan, OPTICS, Density-Based Clustering, EM Algorithm	The paper emphasises the value of K-means clustering in consumer data analysis and business decision-making	[21]
2020	A Comparative Study on K-Means Clustering and Agglomerative Hierarchical Clustering	K-Means and Agglomerative Hierarchy	K-means performs faster for large datasets and agglomerative hierarchical is better for smaller ones.	[22]
2021	Gaussian Mixture Model Clustering with Incomplete Data	GMM	Experiments validate the effectiveness of the proposed algorithm.	[23]
2022	Bayesian Inference-Based Gaussian Mixture Models with Optimal Components Estimation Towards Large-Scale Synthetic Data Generation for In Silico Clinical Trials	BGMM-OCE	BGMM-OCE outperforms other synthetic data generators in terms of computational efficiency and unbiasedness	[24]
2022	Design and Implementation of an Improved K-Means Clustering Algorithm	Improved K-Means	Enhanced algorithm works better than conventional K-Means.	[25]
2022	Gaussian mixture model clustering algorithms for the analysis of high-precision mass measurements	GMM	Results from GMMs were closely congruent with values that had previously been published.	[26]

A. Research Gap

The limited investigation of the Analytical Hierarchy Process (AHP) as a technique for clustering requirements in the context of planning a project's next release is the area of research that will be addressed in this research. Although most of the literature now in existence focuses on the use of AHP in requirements prioritisation and decision-making, there is a striking paucity of studies that explore its potential utility in grouping or clustering requirements to speed up the release planning process. In the context of release planning, AHP in integration with clustering can be used to enhance how requirements are organised, classified, and prioritised. This will ultimately result in more effective and efficient project management.

III. Techniques Used in the Study

A. Requirements Prioritisation Techniques

Software engineering professionals utilise a collection of methodologies called software requirements prioritisation techniques to rank the importance or priority of various software project requirements. Because not all requirements

can be addressed at the same time during software development due to restricted resources (such as time and money), prioritising requirements is essential. To ensure the successful delivery of a software product, it is crucial to identify and concentrate on the most important and significant needs. The two techniques that we will be using in this study are AHP and MoSCoW.

1) Analytical hierarchical process: The Analytical Hierarchy Process (AHP) is a systematic decision-making technique [27] proposed by Thomas L. Saaty in the 1970s. It was developed for complex decision-making so that the decision-maker could set priorities and get to the best option possible [28]. AHP starts by modeling the decision issue as a hierarchical structure and breaks it into three parts: a goal or aim, criteria that help achieve the goal, and alternatives or possibilities that need to be examined. In the next step, experts or decision-makers are requested to compare the criteria and options at each level of the hierarchy in pairs. They utilise a scale to indicate the relative importance of things, often ranging from 1 (equal importance) to 9 (much more essential). Then a consistency check is done to make sure the comparisons are reliable. To determine if decision-makers judgments are consistent, the AHP technique uses mathematical calculations. It may be necessary for decision-makers to reevaluate their conclusions if contradictions are found.

AHP uses pairwise comparison data to determine the relative weights or priorities of the criteria and alternatives. These weights reflect the preference for each choice relative to the criteria and the significance of each criterion in reaching the overall aim. The scores of the options for each criterion are then combined using the estimated weights. Depending on the decision context, different aggregation techniques, such as weighted sum or weighted average, might be used. To rank and evaluate the options based on their overall desirability or performance in relation to the goal, AHP aggregates the aggregated scores. Lastly, decision-makers can use the prioritised rankings and scores to make decisions. Based on the established criteria and their relative relevance, AHP offers an organised and clear way to assess and choose the best alternative.

2) MoSCoW: The Dynamic Software Development Method (DSDM) provides the foundation for the MoSCoW method [29]. It is a common strategy for prioritising requirements. As a matter of fact, it is one of the easiest techniques [30]. The acronym stands for must have, should have, could have, and won't have. The importance or priority of a certain feature within a project is represented by each category. The core project scope is made up of must-haves, which are important and non-negotiable components necessary for project success. Should-haves are crucial characteristics that greatly enhance the value of the project and ought to be applied whenever practical. Could-haves offer flexibility for prospective improvements because they are desired but not necessary. To manage scope and avoid feature creep, won't-haves are expressly left out of the current phase or project. MoSCoW supports resource allocation and project planning by assisting project teams and stakeholders in prioritising requirements, ensuring that critical components are addressed first while providing clarity on what may be postponed or excluded.

B. Clustering Algorithms

The need to find knowledge in multidimensional data is growing since massive volumes of data are being continuously collected today. One of the crucial steps in mining or extracting massive information is data miming. Clustering is the most intriguing area of data mining, which seeks to identify underlying patterns in data and identify some useful subgroups for additional investigation. Each group, or cluster, is made up of things that are dissimilar from those in other groups yet like one another [31].

A total of five clustering algorithms have been used in this paper and each algorithm is briefly discussed in this section.

1) K-Means: In machine learning and data mining, the clustering algorithm K-Means is very famous and frequently employed [32]. It requires the number of clusters to be specified prior to the operation [33]. It seeks to divide a given dataset into the specified number of clusters (K) according to how similar the data points are to one another to maximise certain clustering criteria. K-Means is an iterative technique that minimises the sum of squared distances between data points and the centroids of each cluster to give results. The k-means algorithm is a well-liked clustering technique that minimises clustering error [34].

2) Partition Around Medoids (PAM): The PAM method partitions a distance matrix into a predetermined number of clusters [35]. The goal of PAM is to divide a dataset into a predetermined number of clusters by choosing actual data points, known as medoids, as representatives of the clusters. PAM is meant to work with dissimilarity or distance matrices. Like centroids, medoids are chosen from the actual data points, which makes PAM more resistant to noise and outliers.

3) Agglomerative hierarchical clustering: The process of clustering data points into a hierarchical structure of clusters is called agglomerative hierarchical clustering. Due to the exponential rise of real-world data, hierarchical clustering is crucial for data analytics [36]. In this type of clustering, each item at first represents a separate cluster. The appropriate cluster structure is then created by repeatedly merging clusters until all data points are members of a single cluster, or until a stopping requirement is satisfied. A dendrogram, which is a tree-like structure created because of this procedure, shows the clustering hierarchy visually.

4) Gaussian Mixture Models (GMM): Gaussian Mixture Models (GMMs) are probabilistic models used for modelling complicated data distributions in statistical analysis and machine learning. Much research has been done on it due to its usefulness and efficiency [37]. They presume that a variety of Gaussian (normal) distributions, each with its mean and covariance, were combined to produce the data. These

parameters are intended to be learned, and GMMs estimate the likelihood that data points will belong to each Gaussian component. They are frequently used for tasks like clustering, density estimation, and data generation.

5) BIRCH: Balanced Iterative Reducing and Clustering Using Hierarchies is an effective hierarchical clustering algorithm made for grouping huge datasets. Its key characteristic is to employ low memory resources for high-quality clustering of large-scale data datasets and to only scan datasets once to reduce I/O overhead [38]. A comparable B + tree structure known as a Clustering Feature Tree (CF Tree) is used by Birch to perform clustering [39].

IV. Proposed Methodology

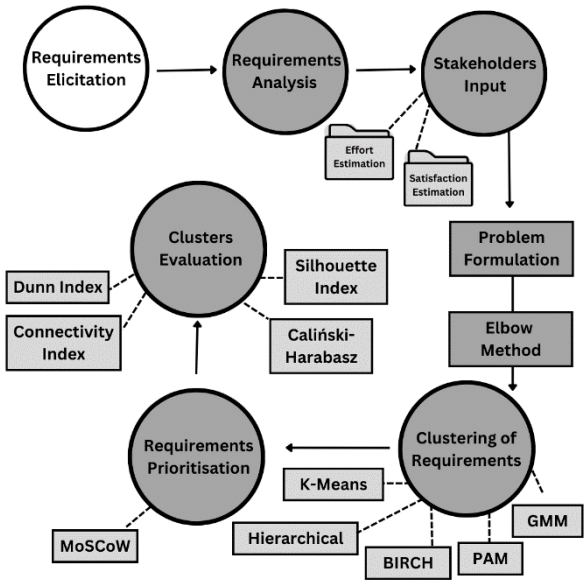

Fig. 1. Proposed methodology.

This study presents a method for prioritizing requirements for the next release using requirements prioritisation methods. It considers the effort required for implementing a requirement and its satisfaction with stakeholders. Clustering algorithms are applied to cluster requirements, and the technique is used to extract a group of requirements for the next release. The validity of clusters is evaluated. In the end MoSCoW is applied to assign importance to the clusters. The Fig. 1 provides a bird's eye view of the process.

A. Requirements Elicitation

Requirement elicitation is a crucial step in requirement engineering, gathering stakeholders' needs and expectations for a software project through discussions, interviews, and surveys, ensuring comprehensive documented requirements.

B. Requirements Analysis

Requirements Analysis involves a thorough examination of requirements to eliminate ambiguity, address inconsistencies, and evaluate feasibility. It aims to create a refined representation of the software's functionalities.

C. Stakeholders' Input

Stakeholders actively contribute to the decision-making process by offering critical input on two important factors: the amount of work necessary to accomplish the project and the expected degree of satisfaction. Their insights cover both effort (resource allocation, time commitments, and potential obstacles) and satisfaction (alignment with organisational goals and client needs). Through the careful balancing of resource optimisation and stakeholder satisfaction throughout project planning, this dual input enables informed decision-making.

D. Problem Formulation

1) Quantitative data: Consider a situation where we have a list of requirements, R = r1, r2 ,..., rn, that reflect the new features that various customers have recommended for a forthcoming software version. Each stakeholder i is given a weight wi to indicate their significance. This implies that some stakeholders' preferences will be taken into consideration more so than others when deciding what issues need to be solved in a software version. The set of customer weights is denoted by the notation W = w1, w2, ..., wn.

Each requirement rj in the set R has a corresponding development effort value ej that calculates the resources or cost necessary for its implementation. The notation for this collection of effort values is E = e1, e2 ,..., en. This is measured by a value vij, which expresses the significance of need rj for customer i. In essence, higher vij values indicate that stakeholder i is given more priority.

Summing up a requirement's importance ratings across all stakeholders yields the total value of including it in the upcoming software release, or its global satisfaction, abbreviated as **sj** (sj = m i=1 wi vij). By considering each stakeholder's own priorities and weights, this indicates the overall satisfaction that the addition of requirement rj would offer to all stakeholders. The set of requirement satisfactions that result is denoted by S = s1, s2,..., sn [40].

2) AHP dataset: For the pairwise comparisons of each criterion in this study, the quantitative data set is used. As a result, the AHP data set for our requirements generated. Following the collection of pairwise comparison judgments, the eigenvector approach is used to determine the respective weights of the two criteria, effort, and satisfaction. Then, a square matrix known as the comparison matrix is formed, with elements $\mathbf{c_{ij}}$ standing in for the weighting of the criteria effort ($\mathbf{c_i}$) and satisfaction ($\mathbf{c_j}$). By dividing each column by its sum, the matrix is normalised, producing a matrix of normalised values. To determine the priority vector for each level, the normalised values in each row are averaged. The consistency ratio (CR), which assesses whether the judgments line coherently, is used in a consistency check to ensure consistent pairwise comparisons. Adjustments are made if the CR exceeds a predetermined limit, which is commonly set at 0.1. The priority vectors show the relative weights of the requirements after the consistency check has been successful.

E. Elbow Method

The elbow method is a heuristic in data science and machine learning for determining the optimal number of

clusters in a dataset. It involves considering a range of potential cluster numbers and computing the sum of squared distances between data points and cluster centers. The study applies the elbow method to the requirements dataset, calculating the within-cluster sum of squares (WCSS) for varying cluster numbers and plotting these values.

F. Clusters Formation

In this phase clusters of requirements are formed. It involves organizing and grouping similar requirements into clusters using techniques like similarity analysis or domain categorization. This process enhances manageability and provides a structured approach for analysis. Five distinct clustering algorithms will be employed: K-Means, Agglomerative Hierarchical Clustering, Partitioning Around Medoids (PAM), Gaussian Mixture Model (GMM), and BIRCH. These algorithms help extract meaningful patterns and structures from requirements, aiding in informed decision-making during the prioritization process.

G. Clusters Evaluation

The evaluation of clusters is crucial for assessing the quality and validity of data analysis or machine learning algorithms. Three mechanisms are used: Dunn Index, Silhouette Index, and Caliński-Harabasz Index, which are calculated after cluster formation and used to rate them.

1) Dunn index: The Dunn Index is a clustering validation statistic that unsupervised machine learning researchers use to rate the accuracy of their clustering findings. It gauges the separation between clusters, or how far away various clusters are, in relation to the compactness of clusters, or how near the data points inside a cluster are to one another. Better clustering with smaller within-cluster distances and larger between-cluster distances is indicated by a higher Dunn Index.

$$\textbf{Dunn Index} = \textbf{min\_intercluster\_distance} / \textbf{max\_intracluster\_distance}$$

Where:

Min_intercluster_distance: The minimal distance between any two centroids that belong to separate clusters.

Max_intracluster_distance: The maximum distance between any two data points within the same cluster.

2) Silhouette index: The Silhouette Index is a tool for clustering evaluation that assesses how cohesive and well-separated clusters are. Higher values denote better clustering quality; the range is -1 to 1. (b(i) - a(i)) / maxa(i), b(i) is the formula for calculating the silhouette score for a single data point, where a(i) is the average distance inside the same cluster and b(i) is the smallest average distance to another cluster. Greater clustering is suggested by average silhouette scores that are higher across all data points.

$$S(i) = (b(i) - a(i)) / \max\{a(i), b(i)\}$$

Where:

S(i): The silhouette score for data point i.

a(i): The average distance between data points i and all other data points in the same cluster.

b(i): The shortest average distance between data point i and all other data points in a distinct cluster.

3) Caliński-Harabasz index: The Calinski-Harabasz Index, commonly referred to as the Variance Ratio Criterion, is a clustering evaluation metric used in unsupervised machine learning to rate the calibre of clusters. It calculates the difference between the variances within and between clusters. Better-defined and more distinct clusters are indicated by higher Calinski-Harabasz Index values.

$$CH = B/W\{(N-K)/(K-1)\}$$

Where:

B: Between-cluster variance, which measures the variance between different clusters.

W: Within-cluster variance, which measures the variance within individual clusters.

N: Total number of data points.

K: Number of clusters.

H. Requirements Prioritisation

The MoSCoW approach is used in this study as a useful tool to prioritise requirements clusters. Requirements clusters are systematically classified and ranked using MoSCoW based on their importance and criticality to the project. This will help in improving efficiency and efficacy of the project planning and resource allocation and will make sure that the development efforts are concentrated on the most important and impactful clusters of needs.

V. Methodology Implementation

A. Formulation of Problem

1) 20 Requirements Problem: There are twenty requirements and five stakeholders in this dataset, and it was drawn from [41]. The level of priority or value assigned by each stakeholder to each requirement is shown in Table II along with the development effort connected to each requirement. The stakeholder weights are offered in the range of 1 to 5 (Table III). These values might be thought of as linguistic terms like "without importance" (1), "less important" (2), "important" (3), "very important" (4), and "extremely important" (5). They also line up with the relative importance of each need. There is an estimated effort score that corresponds to each requirement, ranging from 1 to 10.

TABLE II. 20 Requirements Problem

	C1	*C2*	*C3*	*C4*	*C5*	*Effort*
R1	4	4	5	4	5	1
R2	2	4	3	5	4	4
R3	1	2	3	2	2	2
R4	2	2	3	3	4	3
R5	5	4	4	3	5	4

R6	5	5	5	4	4	7
R7	2	1	2	2	2	10
R8	4	4	4	4	4	2
R9	4	4	4	2	5	1
R10	4	5	4	3	2	3
R11	2	2	2	5	4	2
R12	3	3	4	2	5	5
R13	4	2	1	3	3	8
R14	2	4	5	2	4	2
R15	4	4	4	4	4	1
R16	4	2	1	3	1	4
R17	4	3	2	5	1	10
R18	1	2	3	4	2	4
R19	3	3	3	3	4	8
R20	2	1	2	2	1	4

TABLE III. CUSTOMERS. WEIGHTS FOR 20 REQ. PROBLEM

Customers' Weights	***C1***	***C2***	***C3***	***C4***	***C5***
	1	4	2	3	4

a) 20 Requirements Problem using Quantitative Approach: To convert the data into two dimensions to apply clustering on it, we considered Section 4.4.1. Here:

R = {r1, r2, ……., r20},

E = {1, 4, 2, …….,4},

W = {1, 4, 2, ……., 4}.

This is how 'S' (Satisfaction) was calculated for r1.

$S = \sum (Vij * Wi)$

S= {(4*1) + (4*4) + (5*2) + (4*3) + (5*4)}

S= 62

So, satisfaction for r1 was calculated to be 62 whereas the effort is 1. The rest was also calculated similarly, and this Table IV was generated as a result.

TABLE IV. QUANTITATIVE DATASET FOR 20 REQ. PROBLEM

ID	***Eff.***	***Sat.***	***ID***	***Effort***	***Sat.***
R1	1	62	**R11**	2	45
R2	4	55	**R12**	5	49
R3	2	29	**R13**	8	35
R4	3	41	**R14**	2	50
R5	4	58	**R15**	1	56
R6	7	63	**R16**	4	27
R7	10	24	**R17**	10	39
R8	2	56	**R18**	4	35
R9	1	54	**R19**	4	46
R10	3	49	**R20**	4	20

b) 20 Requirements Problem using AHP: This Table V was created by using the same data set to get the AHP values for effort and satisfaction.

TABLE V. AHP DATASET FOR 20 REQ. PROBLEM

ID	***Effort***	***Satisfaction***
R1	12.7640176	3.24660865
R2	3.19100441	3.65981339
R3	6.38200881	6.9410254
R4	4.25467254	4.90950577
R5	3.19100441	3.4705127
R6	1.82343109	3.19507518
R7	1.27640176	8.38707236
R8	6.38200881	3.59445958
R9	12.7640176	3.72758771
R10	4.25467254	4.10795381
R11	6.38200881	4.47310526
R12	2.55280353	4.10795381
R13	1.5955022	5.75113533
R14	6.38200881	4.02579473
R15	12.7640176	3.59445958
R16	3.19100441	7.45517543
R17	1.27640176	5.1612753
R18	3.19100441	5.75113533
R19	3.19100441	4.37586384
R20	3.19100441	10.0644868

2) 100 Requirements Problem: There are five stakeholders in this data set as well, but there are 100 requirements this time and it was obtained from [42].The difficulty of selecting requirements from a bigger set in the early timeboxes of establishing true agile software projects led to the selection of this dataset. Because of this, we now have 100 requirements rather than simply 20. For the development effort, each requirement has a value that runs from 1 to 20. The maximum development effort in this case is 20 units, or 4 weeks, which roughly corresponds to the timescale set by agile approaches (such as Scrum's proposed iteration length of 2 to 4 weeks). Stakeholders rate the significance of criteria on a scale of 1 to 3. Here, the digits 1-3 stand for (1) not necessary, (2) preferable, or (3) required [43].

The Effort and Satisfaction for each requirement was calculated in the similar way as it was calculated for 20 Requirements problem. The Quantitative and AHP datasets for 100 requirements problem is given in Table VI:

TABLE VI. QUANTITATIVE DATASET (LEFT) AND AHP DATASET (RIGHT) FOR 100 REQ. PROBLEM

ID	***Effort***	***Satisfaction***
R1	16	29
R2	19	23

ID	***Effort***	***Satisfaction***
R1	0.35245612	0.87906114
R2	0.29680515	1.10838143

R3	16	18
R4	7	21
R5	19	22
R6	15	20
R7	8	22
R8	10	29
R9	6	27
R10	18	21
R11	15	31
R12	12	33
R13	16	33
R14	20	25
R15	9	25
R16	4	30
R17	16	25
R18	2	28
R19	9	35
R20	3	29
R21	2	27
R22	10	23
R23	4	28
R24	2	29
R25	7	36
R26	15	28
R27	8	30
R28	20	22
R29	9	30
R30	11	32
R31	5	20
R32	1	31
R33	17	24
R34	6	26
R35	2	24
R36	16	23
R37	8	26
R38	12	32
R39	18	26
R40	5	27
R41	6	32
R42	14	30
R43	15	15
R44	20	26
R45	14	29
R46	9	28
R47	16	27

R3	0.35245612	1.41626516
R4	0.80561398	1.21394157
R5	0.29680515	1.15876241
R6	0.37595319	1.27463865
R7	0.70491224	1.15876241
R8	0.56392979	0.87906114
R9	0.93988298	0.94417678
R10	0.31329433	1.21394157
R11	0.37595319	0.82234751
R12	0.46994149	0.77250827
R13	0.35245612	0.77250827
R14	0.28196489	1.01971092
R15	0.62658865	1.01971092
R16	1.40982447	0.8497591
R17	0.35245612	1.01971092
R18	2.81964894	0.91045618
R19	0.62658865	0.72836494
R20	1.87976596	0.87906114
R21	2.81964894	0.94417678
R22	0.56392979	1.10838143
R23	1.40982447	0.91045618
R24	2.81964894	0.87906114
R25	0.80561398	0.70813258
R26	0.37595319	0.91045618
R27	0.70491224	0.8497591
R28	0.28196489	1.15876241
R29	0.62658865	0.8497591
R30	0.51266344	0.79664915
R31	1.12785958	1.27463865
R32	5.63929788	0.82234751
R33	0.3317234	1.06219887
R34	0.93988298	0.98049127
R35	2.81964894	1.06219887
R36	0.35245612	1.10838143
R37	0.70491224	0.98049127
R38	0.46994149	0.79664915
R39	0.31329433	0.98049127
R40	1.12785958	0.94417678
R41	0.93988298	0.79664915
R42	0.40280699	0.8497591
R43	0.37595319	1.6995182
R44	0.28196489	0.98049127
R45	0.40280699	0.87906114
R46	0.62658865	0.91045618
R47	0.35245612	0.94417678

R48	6	21
R49	6	28
R50	6	32
R51	6	34
R52	2	27
R53	17	24
R54	18	30
R55	1	24
R56	3	35
R57	14	35
R58	16	18
R59	18	23
R60	7	26
R61	10	18
R62	7	28
R63	16	29
R64	19	38
R65	17	25
R66	15	22
R67	11	23
R68	8	26
R69	20	34
R70	1	15
R71	5	23
R72	8	32
R73	3	28
R74	15	29
R75	4	21
R76	20	21
R77	10	31
R78	20	39
R79	3	21
R80	20	23
R81	10	22
R82	16	22
R83	19	24
R84	3	25
R85	12	29
R86	16	15
R87	15	28
R88	1	21
R89	6	34
R90	7	32
R91	15	27
R92	18	32

R48	0.93988298	1.21394157
R49	0.93988298	0.91045618
R50	0.93988298	0.79664915
R51	0.93988298	0.74978744
R52	2.81964894	0.94417678
R53	0.3317234	1.06219887
R54	0.31329433	0.8497591
R55	5.63929788	1.06219887
R56	1.87976596	0.72836494
R57	0.40280699	0.72836494
R58	0.35245612	1.41626516
R59	0.31329433	1.10838143
R60	0.80561398	0.98049127
R61	0.56392979	1.41626516
R62	0.80561398	0.91045618
R63	0.35245612	0.87906114
R64	0.29680515	0.67086245
R65	0.3317234	1.01971092
R66	0.37595319	1.15876241
R67	0.51266344	1.10838143
R68	0.70491224	0.98049127
R69	0.28196489	0.74978744
R70	5.63929788	1.6995182
R71	1.12785958	1.10838143
R72	0.70491224	0.79664915
R73	1.87976596	0.91045618
R74	0.37595319	0.87906114
R75	1.40982447	1.21394157
R76	0.28196489	1.21394157
R77	0.56392979	0.82234751
R78	0.28196489	0.65366084
R79	1.87976596	1.21394157
R80	0.28196489	1.10838143
R81	0.56392979	1.15876241
R82	0.35245612	1.15876241
R83	0.29680515	1.06219887
R84	1.87976596	1.01971092
R85	0.46994149	0.87906114
R86	0.35245612	1.6995182
R87	0.37595319	0.91045618
R88	5.63929788	1.21394157
R89	0.93988298	0.74978744
R90	0.80561398	0.79664915
R91	0.37595319	0.94417678
R92	0.31329433	0.79664915

R93	4	27
R94	7	25
R95	2	21
R96	7	24
R97	8	24
R98	7	39
R99	7	18
R100	3	27

R93	1.40982447	0.94417678
R94	0.80561398	1.01971092
R95	2.81964894	1.21394157
R96	0.80561398	1.06219887
R97	0.70491224	1.06219887
R98	0.80561398	0.65366084
R99	0.80561398	1.41626516
R100	1.87976596	0.94417678

B. Determining No. of Clusters

To determine the ideal number of clusters, the elbow approach was used on data sets from the 20 and 100 Requirements Problem. The ideal number of clusters is depicted in Fig. 2 and 3.

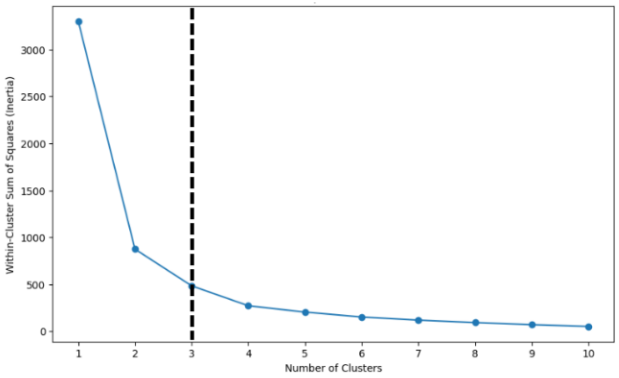

Fig. 2. Optimum no. of clusters using AHP dataset for 20 req. problem.

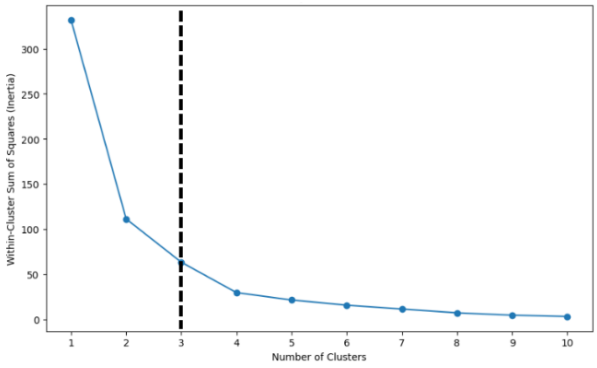

Fig. 3. Optimum no. of clusters using AHP Dataset for 100 req. problem.

C. Clusters Formation and Evaluation

The elbow method's findings show that three clusters are the ideal number for both the 20 and 100 Requirements Problems. We made 3 and 4 clusters because we are employing MoSCoW in addition to AHP for requirement prioritising. This is because MoSCoW has four characteristics.

In the publication [40], quantitative dataset was used to evaluate three clustering algorithms: K-means, Hierarchical Clustering, and Partition Around Medoids (PAM). In this research, we compare the values acquired by the Analytic Hierarchy Process (AHP) approach to the values of quantitative dataset. The benefits and drawbacks of various techniques are better understood through holistic comparison, which also advances knowledge of efficient clustering methodologies and their real-world applications.

The graphical depiction of 100 requirements datasets for Agglomerative Hierarchical Clustering is illustrated in Fig. 4 and 5. This visualization provides a clear representation of the analyzed data, offering insights into the observed trends and patterns.

To gain a deeper knowledge of how the proposed technique interacts with various clustering algorithms, evaluation indices for both types of data sets, namely Quantitative and AHP, are also calculated using Gaussian Mixture Models (GMM) and BIRCH (Fig. 6 and 7).

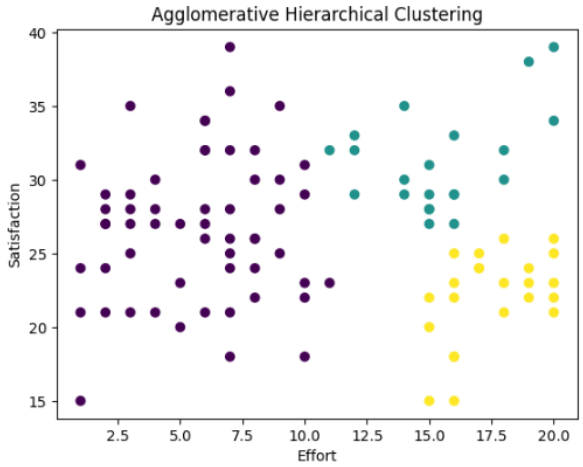

Fig. 4. Hierarchical clustering for 100 req. problem using quantitative dataset.

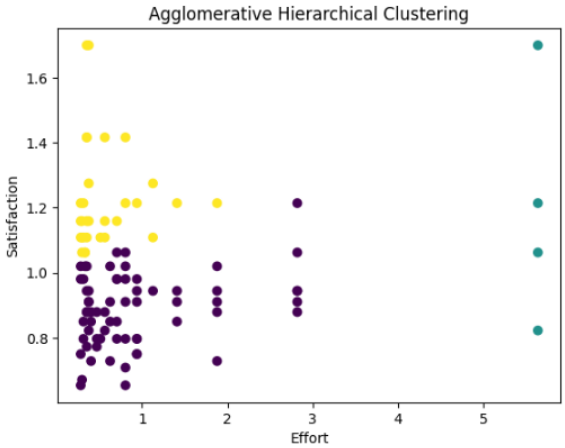

Fig. 5. Hierarchical clustering for 100 req. problem using AHP dataset.

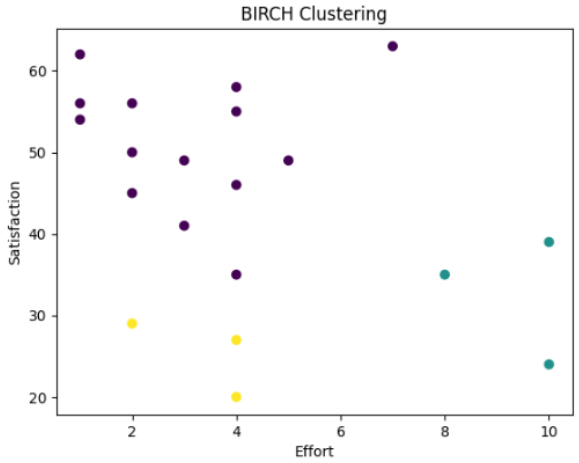

Fig. 6. BIRCH clustering for 20 req. problem using quantitative dataset.

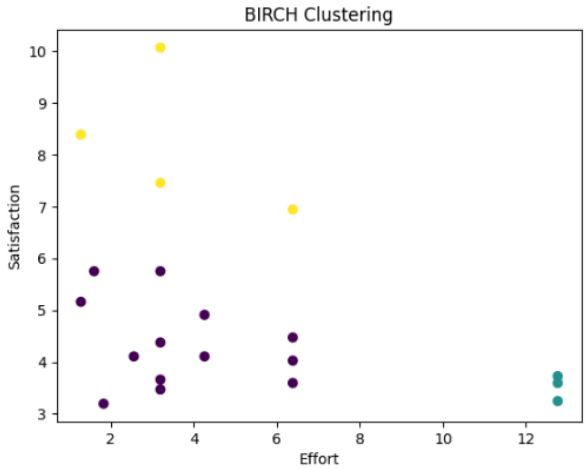

Fig. 7. BIRCH clustering for 20 req. problem using AHP dataset.

All in all, five clustering algorithms: K-Means, Partition Around Medoids, Agglomerative Hierarchical Clustering, Gaussian Mixture Models, and BIRCH and three evaluation metrics: the Dunn Index, the Silhouette Index, and the Calinski-Harabasz Index are used in this research.

The outcomes of each clustering algorithm for cluster evaluation metrics are provided in the Tables VII-XVI.

1) K-Means
2) PAM
3) Hierarchical
4) GMM
5) BIRCH

TABLE VII. EVALUATION METRICS FOR 20 REQ. PROBLEM

20 Requirements Problem			
	Clusters	***Quantitative***	***AHP***
Dunn	3	0.209	0.4336
Silhouette	3	0.4666	0.5690
CH	3	22.9273	33.7443
Dunn	4	0.2527	0.2417
Silhouette	4	0.4176	0.4863
CH	4	24.3832	34.1044

TABLE VIII. EVALUATION METRICS FOR 100 REQ. PROBLEM

100 Requirements Problem			
	Clusters	***Quantitative***	***AHP***
Dunn	3	0.0548	**0.2364**
Silhouette	3	0.4283	**0.4632**
CH	3	89.5132	**89.7174**
Dunn	4	0.0783	**0.2377**
Silhouette	4	0.3993	**0.4766**
CH	4	90.9959	**96.8018**

TABLE IX. EVALUATION METRICS FOR 20 REQ. PROBLEM

20 Requirements Problem			
	Clusters	***Quantitative***	***AHP***
Dunn	3	0.2607	**2.7100**
Silhouette	3	0.4843	**0.5208**
CH	3	22.6144	**31.1727**
Dunn	4	0.3151	**1.5103**
Silhouette	4	0.4116	**0.4374**
CH	4	24.0329	**31.2174**

TABLE X. EVALUATION METRICS FOR 100 REQ. PROBLEM

100 Requirements Problem			
	Clusters	***Quantitative***	***AHP***
Dunn	3	0.0831	**0.3396**
Silhouette	3	0.4308	**0.3943**
CH	3	**89.5132**	46.9101
Dunn	4	0.0696	**0.3024**
Silhouette	4	0.3993	**0.3998**
CH	4	**88.7641**	64.6714

TABLE XI. EVALUATION METRICS FOR 20 REQ. PROBLEM

20 Requirements Problem			
	Clusters	***Quantitative***	***AHP***
Dunn	3	0.2576	**2.9804**
Silhouette	3	0.4549	**0.5690**
CH	3	18.6832	**33.7443**
Dunn	4	0.2482	**2.7427**

Silhouette	4	0.3561	**0.4863**
CH	4	18.7909	**34.1044**

TABLE XII. EVALUATION METRICS FOR 100 REQ. PROBLEM

100 Requirements Problem			
	Clusters	***Quantitative***	***AHP***
Dunn	3	0.1096	**0.3472**
Silhouette	3	0.4278	**0.4327**
CH	3	88.0933	**82.8722**
Dunn	4	0.1096	**0.2518**
Silhouette	4	0.3964	**0.4576**
CH	4	82.5902	**95.1834**

TABLE XIII. EVALUATION METRICS FOR 20 REQ. PROBLEM

20 Requirements Problem			
	Clusters	***Quantitative***	***AHP***
Dunn	3	0.2739	**0.3723**
Silhouette	3	0.4568	**0.5690**
CH	3	22.5821	**33.744**
Dunn	4	0.1796	**0.310**
Silhouette	4	0.3839	**0.4905**
CH	4	22.0866	**33.633**

TABLE XIV. EVALUATION METRICS FOR 100 REQ. PROBLEM

100 Requirements Problem			
	Clusters	***Quantitative***	***AHP***
Dunn	3	**0.7259**	0.1706
Silhouette	3	**0.4285**	0.0743
CH	3	**90.674**	26.5032
Dunn	4	**0.5557**	0.077
Silhouette	4	**0.3721**	0.1082
CH	4	**90.7001**	36.2847

TABLE XV. EVALUATION METRICS FOR 20 REQ. PROBLEM

20 Requirements Problem			
	Clusters	***Quantitative***	***AHP***
Dunn	3	**12.9526**	7.249
Silhouette	3	0.4672	**0.5690**
CH	3	18.9442	**33.744**

TABLE XVI. EVALUATION METRICS FOR 100 REQ. PROBLEM

100 Requirements Problem			
	Clusters	***Quantitative***	***AHP***
Dunn	3	**8.9139**	0.665
Silhouette	3	**0.4384**	0.4053
CH	3	**96.1607**	79.1779

D. Prioritisation of Requirements

The MoSCoW method is used to prioritize requirements clusters. Clusters with higher satisfaction and minimal effort were given the highest priority and are designated as "MUST" fulfillments. Clusters with higher satisfaction and minimal effort are designated as "SHOULD" requirements. Clusters in the "COULD" category are considered for enhancement due to their higher effort cost. Clusters in the "WON'T" category are intentionally deferred due to higher effort requirements. This dynamic prioritization methodology offers a nuanced perspective for optimizing software requirements in line with project goals.

VI. DISCUSSION

Our study compared the Analytic Hierarchy Process (AHP) with quantitative dataset approaches in requirement prioritization and clustering, highlighting performance differences across multiple evaluations. Each comparison table illustrates instances where either AHP or the quantitative dataset method performed better, with the superior values highlighted for clarity Table (VII-XVI). Out of 54 evaluations, AHP showed superior performance in 39 cases, emphasizing variability between methods.

The effectiveness of AHP in generating compact and meaningful clusters underscores its potential for handling complex datasets in software engineering. By leveraging a structured decision-making approach that incorporates both qualitative and quantitative judgments, AHP successfully groups requirements with closer features or similarities together more cohesively. This results in coherent and relevant requirement groupings, which in turn facilitates improved decision-making and prioritization within software development processes. AHP's ability to create compact clusters highlights its utility in enhancing the efficiency and effectiveness of software engineering practices.

VII. RESULTS

The Analytic Hierarchy Process (AHP) and the quantitative datasets were compared 54 times in total using evaluation metrics. The purpose of these comparisons was to assess the efficiency and performance of the AHP approach in comparison to the quantitative data representation. 39 of these 54 comparisons revealed that the AHP technique performed better than other approaches. This indicates that, in contrast to the quantitative data technique, AHP typically produced more favorable outcomes or results.

This finding's relevance stems from the AHP approach's consistent propensity to outperform the quantitative data representation over a sizable majority of the comparisons. This series of outcomes highlights the possible advantages of applying the AHP approach to cluster or analyse the provided dataset, suggesting that it might be a more efficient and reliable method for producing valuable insights or groups.

VIII. Conclusion and Future Work

The importance of using data mining techniques to efficiently prioritise requirements in software engineering is shown by this study. It also emphasises the extraordinary excellence of the Analytic Hierarchy Process (AHP) in the context of software engineering for prioritising software requirements. Based on a detailed analysis of five clustering algorithms and three cluster assessment indices, our results consistently demonstrate that AHP outperforms traditional quantitative data representations in the majority of the 54 comparisons conducted. Furthermore, the combination of AHP with the MoSCoW needs prioritisation framework not only led to better results but also enhanced resource allocation, flexible planning, and increased stakeholder satisfaction. This study recommends using AHP, data mining techniques, and the MoSCoW framework as the suggested methodology for prospective projects.

Since the data sets were generated manually with the help of stakeholders in this research. In the future, we can use machine learning algorithms. These algorithms can be trained on historical project data to learn the underlying patterns and characteristics of similar projects. By improving the overall efficiency of requirements prioritisation techniques, this integration could pave the way for more sophisticated and context-sensitive approaches to managing software requirements.

References

[1] P. Achimugu, A. Selamat, R. Ibrahim, and M. N. Mahrin, "A systematic literature review of software requirements prioritization research," Inf Softw Technol, vol. 56, no. 6, pp. 568–585, Jun. 2014, doi: 10.1016/j.infsof.2014.02.001.

[2] X. Franch and G. Ruhe, "Software release planning," in Proceedings of the 38th International Conference on Software Engineering Companion, New York, NY, USA: ACM, May 2016, pp. 894–895. doi: 10.1145/2889160.2891051.

[3] M. Azzolini and L. I. Passoni, "Prioritization of Software Requirements: a Cognitive Approach," in Procedings of the Fourth International Workshop on Knowledge Discovery, Knowledge Management and Decision Support, Paris, France: Atlantis Press, 2013. doi: 10.2991/.2013.13.

[4] I. Olaronke, I. Rhoda, and G. Ishaya, "An Appraisal of Software Requirement Prioritization Techniques," Asian Journal of Research in Computer Science, pp. 1–16, Apr. 2018, doi: 10.9734/ajrcos/2018/v1i124717.

[5] K. El Emam and A. G. Koru, "A Replicated Survey of IT Software Project Failures," IEEE Softw, vol. 25, no. 5, pp. 84–90, Sep. 2008, doi: 10.1109/MS.2008.107.

[6] A. Ahmad, M. Goransson, and A. Shahzad, "Limitations of the Analytic Hierarchy Process Technique with Respect to Geographically Distributed Stakeholders," World Acad Sci Eng Technol, pp. 111–116, 2010.

[7] P. Govender and V. Sivakumar, "Application of k-means and hierarchical clustering techniques for analysis of air pollution: A review (1980–2019)," Atmos Pollut Res, vol. 11, no. 1, pp. 40–56, Jan. 2020, doi: 10.1016/j.apr.2019.09.009.

[8] M. Kassab and N. Kilicay-Ergin, "Applying analytical hierarchy process to system quality requirements prioritization," Innov Syst Softw Eng, vol. 11, no. 4, pp. 303–312, Dec. 2015, doi: 10.1007/s11334-015-0260-8.

[9] J. Ali Khan, I. Ur Rehman, Y. Hayat Khan, I. Javed Khan, and S. Rashid, "Comparison of Requirement Prioritization Techniques to Find Best Prioritization Technique," International Journal of Modern Education and Computer Science, vol. 7, no. 11, pp. 53–59, Nov. 2015, doi: 10.5815/ijmecs.2015.11.06.

[10] J. A. Khan, Izaz-ur-Rehman, S. P. Khan, I. Qasim, and Y. H. Khan, "An Evaluation of Requirement Prioritization Techniques with ANP," International Journal of Advanced Computer Science and Applications, vol. 7, no. 7, 2016.

[11] S. Parthasarathy and M. Daneva, "An approach to estimation of degree of customization for ERP projects using prioritized requirements," Journal of Systems and Software, vol. 117, pp. 471–487, Jul. 2016, doi: 10.1016/j.jss.2016.04.006.

[12] K. S. Ahmad, N. Ahmad, H. Tahir, and S. Khan, "Fuzzy_MoSCoW: A fuzzy based MoSCoW method for the prioritization of software requirements," in 2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT), IEEE, Jul. 2017, pp. 433–437. doi: 10.1109/ICICICT1.2017.8342602.

[13] M. S. Jahan, F. Azam, M. W. Anwar, A. Amjad, and K. Ayub, "A Novel Approach for Software Requirement Prioritization," in 2019 7th International Conference in Software Engineering Research and Innovation (CONISOFT), IEEE, Oct. 2019, pp. 1–7. doi: 10.1109/CONISOFT.2019.00012.

[14] M. Yaseen, A. Mustapha, and N. Ibrahim, "Prioritization of Software Functional Requirements from Developers Perspective," International Journal of Advanced Computer Science and Applications, vol. 11, no. 9, 2020.

[15] N. Mohamed, S. Mazen, and W. Helmy, "E-AHP: An Enhanced Analytical Hierarchy Process Algorithm for Priotrizing Large Software Requirements Numbers," International Journal of Advanced Computer Science and Applications, vol. 13, no. 7, 2022, doi: 10.14569/IJACSA.2022.0130725.

[16] A. Bouguettaya, Q. Yu, X. Liu, X. Zhou, and A. Song, "Efficient agglomerative hierarchical clustering," *Expert Syst Appl*, vol. 42, no. 5, pp. 2785–2797, Apr. 2015, doi: 10.1016/j.eswa.2014.09.054.

[17] F. Fouedjio, "A hierarchical clustering method for multivariate geostatistical data," Spat Stat, vol. 18, pp. 333–351, Nov. 2016, doi: 10.1016/j.spasta.2016.07.003.

[18] Z. Li, G. Wang, and G. He, "Milling tool wear state recognition based on partitioning around medoids (PAM) clustering," The International Journal of Advanced Manufacturing Technology, vol. 88, no. 5–8, pp. 1203–1213, Feb. 2017, doi: 10.1007/s00170-016-8848-1.

[19] G. Pitolli, L. Aniello, G. Laurenza, L. Querzoni, and R. Baldoni, "Malware family identification with BIRCH clustering," in 2017 International Carnahan Conference on Security Technology (ICCST), IEEE, Oct. 2017, pp. 1–6. doi: 10.1109/CCST.2017.8167802.

[20] K. P. Sinaga and M.-S. Yang, "Unsupervised K-Means Clustering Algorithm," IEEE Access, vol. 8, pp. 80716–80727, 2020, doi: 10.1109/ACCESS.2020.2988796.

[21] M. Faizan, M. F., S. Ismail, and S. Sultan, "Applications of Clustering Techniques in Data Mining: A Comparative Study," International Journal of Advanced Computer Science and Applications, vol. 11, no. 12, 2020, doi: 10.14569/IJACSA.2020.0111218.

[22] K. B, "A Comparative Study on K-Means Clustering and Agglomerative Hierarchical Clustering," International Journal of Emerging Trends in Engineering Research, vol. 8, no. 5, pp. 1600–1604, May 2020, doi: 10.30534/ijeter/2020/20852020.

[23] Y. Zhang et al., "Gaussian Mixture Model Clustering with Incomplete Data," ACM Transactions on Multimedia Computing, Communications, and Applications, vol. 17, no. 1s, pp. 1–14, Jan. 2021, doi: 10.1145/3408318.

[24] V. C. Pezoulas, N. S. Tachos, G. Gkois, I. Olivotto, F. Barlocco, and D. I. Fotiadis, "Bayesian Inference-Based Gaussian Mixture Models With Optimal Components Estimation Towards Large-Scale Synthetic Data Generation for In Silico Clinical Trials," IEEE Open J Eng Med Biol, vol. 3, pp. 108–114, 2022, doi: 10.1109/OJEMB.2022.3181796.

[25] H. Zhao, "Design and Implementation of an Improved K-Means Clustering Algorithm," Mobile Information Systems, vol. 2022, pp. 1–10, Sep. 2022, doi: 10.1155/2022/6041484.

[26] C. M. Weber, D. Ray, A. A. Valverde, J. A. Clark, and K. S. Sharma, "Gaussian mixture model clustering algorithms for the analysis of high-

precision mass measurements," *Nucl Instrum Methods Phys Res A*, vol. 1027, p. 166299, Mar. 2022, doi: 10.1016/j.nima.2021.166299.

[27] B. Regnell, M. Höst, J. N. och Dag, P. Beremark, and T. Hjelm, "An Industrial Case Study on Distributed Prioritisation in Market-Driven Requirements Engineering for Packaged Software," *Requir Eng*, vol. 6, no. 1, pp. 51–62, Feb. 2001, doi: 10.1007/s007660170015.

[28] Bruce L. Golden, Edward A. Wasil, and Patrick T. Harker, Eds., *"The analytic hierarchy process." Applications and Studies*. Heidelberg, 1989.

[29] S. Hatton, "Early Prioritisation of Goals," in *Advances in Conceptual Modeling – Foundations and Applications*, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 235–244. doi: 10.1007/978-3-540-76292-8_29.

[30] A. Hudaib, R. Masadeh, M. H. Qasem, and A. Alzaqebah, "Requirements Prioritization Techniques Comparison," *Mod Appl Sci*, vol. 12, no. 2, p. 62, Jan. 2018, doi: 10.5539/mas.v12n2p62.

[31] Pradeep Rai and Shubha Singh, "A Survey of Clustering Techniques," *Int J Comput Appl*, vol. 7, Oct. 2010.

[32] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A K-Means Clustering Algorithm," *Appl Stat*, vol. 28, no. 1, p. 100, 1979, doi: 10.2307/2346830.

[33] K. P. Sinaga and M.-S. Yang, "Unsupervised K-Means Clustering Algorithm," *IEEE Access*, vol. 8, pp. 80716–80727, 2020, doi: 10.1109/ACCESS.2020.2988796.

[34] A. Likas, N. Vlassis, and J. J. Verbeek, "The global k-means clustering algorithm," *Pattern Recognit*, vol. 36, no. 2, pp. 451–461, Feb. 2003, doi: 10.1016/S0031-3203(02)00060-2.

[35] L Kaufman and PJ Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons., 2009.

[36] A. Bouguettaya, Q. Yu, X. Liu, X. Zhou, and A. Song, "Efficient agglomerative hierarchical clustering," *Expert Syst Appl*, vol. 42, no. 5, pp. 2785–2797, Apr. 2015, doi: 10.1016/j.eswa.2014.09.054.

[37] Y. Zhang *et al.*, "Gaussian Mixture Model Clustering with Incomplete Data," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 17, no. 1s, pp. 1–14, Jan. 2021, doi: 10.1145/3408318.

[38] K. Peng, L. Zheng, X. Xu, T. Lin, and V. C. M. Leung, "Balanced Iterative Reducing and Clustering Using Hierarchies with Principal Component Analysis (PBirch) for Intrusion Detection over Big Data in Mobile Cloud Environment," 2018, pp. 166–177. doi: 10.1007/978-3-030-05345-1_14.

[39] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH," *ACM SIGMOD Record*, vol. 25, no. 2, pp. 103–114, Jun. 1996, doi: 10.1145/235968.233324.

[40] J. del Sagrado and I. M. del Águila, "Assisted requirements selection by clustering," *Requir Eng*, vol. 26, no. 2, pp. 167–184, Jun. 2021, doi: 10.1007/s00766-020-00341-1.

[41] D. Greer and G. Ruhe, "Software release planning: an evolutionary and iterative approach," *Inf Softw Technol*, vol. 46, no. 4, pp. 243–253, Mar. 2004, doi: 10.1016/j.infsof.2003.07.002.

[42] J. del Sagrado, I. M. del Águila, and F. J. Orellana, "Multi-objective ant colony optimization for requirements selection," *Empir Softw Eng*, vol. 20, no. 3, pp. 577–610, Jun. 2015, doi: 10.1007/s10664-013-9287-3.

[43] E. Simmons, "Requirements triage: what can we learn from a 'medical' approach?," *IEEE Softw*, vol. 21, no. 4, pp. 86–88, Jul. 2004, doi: 10.1109/MS.2004.25.