Journal of Information Systems and Informatics
Vol. 5, No. 3, September 2023 e-ISSN: 2656-4882 p-ISSN: 2656-5935
DOI: 10.51519/journalisi.v5i3.561 Published By DRPM-UBD

Modified Genetic Algorithm and Association Rule Mining for the Retail Sector

Piyush Vyas[1], Aditya Nagdiya [2]

[1] Texas A&M University-Central Texas, Texas, United State
[2]Devi Ahilya Vishwavidyalaya, Madhya Pradesh, India
Email: erpiyush031@gmail.com [1], aditya22.ad@gmail.com[2]

Abstract

This paper concentrates on the optimization of elementary association rule mining. The basic approach of association rule mining generates the positive association rule but focusing on both positive and negative association rule mining to find out efficient results is lacking. Thus our aim is to provide an approach to optimize all positive and negative association rules with the help of a modified genetic algorithm. A genetic algorithm is an optimization technique that provides the best possible solutions that are stronger than the other solutions. The present approach focuses on the importance of population through mean fitness value for further genetic algorithm operation. This paper also shows a comparison between normal Apriori, the Genetic Algorithm, and our proposed algorithm. Where in as a result the proposed approach worked better than others. We believe that the proposed methodology would increase the efficiency of the Decision support system of retail stores.

Keywords: Association Rule Mining, Apriori Algorithm, Genetic algorithm, Data Mining.

1. INTRODUCTION

Data mining is a process employed to autonomously derive knowledge from datasets. Within the realm of data mining, several major domains include Association Rule Mining, Classification, and Clustering. Among these, Association Rule Mining (ARM) has notably garnered significant attention in the research community. It involves the extraction of valuable insights from extensive databases by generating rules. This procedure comprises two fundamental stages: firstly, identifying frequent item sets within the database based on a specified support threshold, and secondly, constructing association rules from these frequent item sets while ensuring a defined level of confidence. The initial step, the discovery of frequent item sets, is particularly computationally intensive as it involves handling an exponentially growing number of item sets in relation to the total number of items. Over the years, numerous efficient algorithms have been developed to address this challenge and facilitate the mining of frequent item sets. [1], [2].

There are a lot of data mining algorithms that found positive rules but a lot of work is done on positive rules and now researchers are working on the negative association rules or infrequent item sets. Thus, our aim is to work on both kind of item set either frequent or infrequent. Due to our results after applying Apriori, we got all kinds of positive and negative rules. Suppose we got $A \Rightarrow B$ which implies that if A then B means it is the positive rule and if we got $\neg A \Rightarrow B$, $A \Rightarrow \neg B$, $\neg A \Rightarrow \neg B$ it means these all are negative rules. We treat these rules as true positive, true negative, false positive, false negative [3], [4].

As we all know genetic algorithm is a search algorithm that provides optimal solutions. After applying the Apriori algorithm [5], we got a lot of rules in which some rules are important and some are not so, to reduce the number of rules and get an important association rule we apply the genetic algorithm for optimization. The genetic algorithm provides the global optimum solution for association rule mining. First, we take some experimental datasets and apply the Apriori algorithm to them. We got frequent item sets after passing the minimum support condition. After getting all the frequent items we put them as input in the genetic algorithm and compute their fitness values. After applying all genetic operators we got finally optimized rules that are strong and less in number. Now we apply our modified genetic algorithm over the same generated rules and get more efficient and less no. of rules than normal genetics.

In this research paper, we next discuss positive and negative association rules in section 2 then we discuss genetic algorithm operators in section 3. In section 4 we discuss our modified genetic algorithm. Finally, we discuss experimental datasets, results, and comparisons in section 5.

2. POSITIVE AND NEGATIVE ASSOCIATION RULES

Association Rule Mining (ARM) is a process directed at identifying prevalent patterns, relationships, or causal connections within a collection of items contained within transactional databases or similar data repositories. ARM's primary objective is to identify groups of item subsets or attributes that frequently appear together in numerous records or transactions. Moreover, it seeks to derive rules outlining the influence of one subset of items on the presence or occurrence of another subset. ARM algorithms are instrumental in revealing predictive rules of a higher order, typically expressed as follows: when certain conditions based on the values of predictive attributes hold true, one can forecast values for specific goal attributes [3], [6].

To generate association rules we have to know about support and confidence through which we got all rules from frequent and infrequent item sets. Basically, support is a probability of occurrence means how many times an item occurs in

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

transactions. Suppose we have T = {t1, t2. . . tn} transactional database of n records with a set of items I = {i1, i2, . . . , im}. The rule A⇒B has a support (denoted as supp) s in the database DB if s% of the transactions in DB contains A∪B. In other words, the support of the rule is the probability that A and B hold together among all the possible presented cases i.e.,

$$\text{Supp}(A \Rightarrow B) = \text{supp}(A \cup B) = P(A \cup B) \quad (1)$$

Here the support of rule A⇒B is the support of A∪B, where A∪B means both A and B occur at the same time in the same transaction. For example a database, D consisting 9 transactions. Suppose min. support count required is 2 (i.e. min_sup = 2/9 = 22 %) [7]–[9].

So, the confidence of a rule A⇒B is,

$$\text{Conf}(A \Rightarrow B) = \text{Supp}(A \cup B)/\text{Supp}(A) \quad (2)$$

Now we discuss the positive association rule, As we previously wrote generally research is done over frequent item sets. Suppose we take an example of a retail shop "If a customer buy milk then he/she also buy bread" so the rule stands as A⇒B. These kinds of rule are known as positive rules. But what happened to these kinds of rules like, "If a customer buys tea he/she doesn't buy coffee." Generally, these rules are known as negative rules like ¬A⇒B, A⇒¬B, ¬A ⇒¬B. Here some negative rules are defined and their support and confidence calculations are also defined, First is the subsequent negative Rule in,

$$\text{Supp}(\neg A => B) = \text{supp}(B) - \text{supp}(A \cup B) \quad (3)$$
$$\text{Conf}(\neg A => B) = \text{supp}(B) - \text{supp}(A \cup B)/1 - \text{supp}(A) \quad (4)$$

Second is the Antecedent Negative Rule in it,

$$\text{Supp}(A => \neg B) = \text{supp}(A) - \text{supp}(A \cup B) \quad (5)$$
$$\text{Conf}(A => \neg B) = \text{supp}(A) - \text{supp}(A \cup B)/\text{supp}(A) \quad (6)$$

The last one is Antecedent and Consequent Negative in it,

$$\text{Supp}(\neg A => \neg B) = 1 - \text{supp}(A) - \text{supp}(B) + \text{supp}(A \cup B) \quad (7)$$
$$\text{Conf}(\neg A => \neg B) = 1 - \text{supp}(A) - \text{supp}(B) + \text{supp}(A \cup B)/1 - \text{supp}(A) \quad (8)$$

The negative association rules discovery seeks rules of the three forms with their support and confidence greater than, lesser than, or equal to, user-specified min_supp and min_conf thresholds respectively [8], [9].

3. GENETIC ALGORITHM

The Genetic Algorithm (GA) is a computational approach that amalgamates Charles Darwin's evolutionary theory with John Holland's work on sexual reproduction in 1970. GA operates on a stochastic search framework, emulating the principles of natural selection. Its versatile application extends to a multitude of domains including artificial intelligence, optimization, and machine learning. GA's iterative nature plays a pivotal role in generating fresh populations of strings from preexisting ones. These strings, or chromosomes, act as binary-encoded representations of potential solutions. Each string undergoes evaluation through a fitness function as part of the problem-solving process. Key components of GA, namely Selection, Crossover, and Mutation, collectively contribute to the creation of an entirely new generation from an initially random population [10]. Following are the operators within the genetic algorithm.

2.1 Binary Coding

Binary coding stands as the preferred method for representing individual genes. It encompasses encoding techniques involving bits, numbers, trees, arrays, lists, or other data structures. Binary encoding is predominantly favored, providing numerous possible chromosomes with fewer alleles. However, it may not align naturally with every problem, necessitating post-genetic operation corrections. The common practice employs binary strings composed of 1s and 0s, with string length contingent upon the desired accuracy [11].

2.2 Random Selection

Selection serves as the vital process for choosing two parent chromosomes from the initial population for crossover. Beyond selecting an encoding method, the subsequent step involves determining the selection procedure, i.e., how individuals within the population are chosen to produce offspring for the succeeding generation, and the number of offspring created per selected parent pair. Random Selection is a technique that randomly selects chromosomes based on their fitness function evaluations, also referred to as the fitness function. Selection pressure quantifies the extent to which superior individuals are favored, greatly influencing GA's convergence rate, with higher selection pressures resulting in quicker convergence [11], [12].

2.3 Crossover

A variety of crossover techniques are at the disposal of GA, including Single-point, Two-point, Uniform, and half-uniform crossovers, as well as three-parent crossover and Crossover for ordered chromosomes. Essentially, the crossover operation selects a random gene along the chromosome's length and swaps all

genes beyond that point. In essence, crossover combines the solutions of two-parent entities to produce offspring, enriching the population with superior individuals. The reproduction process duplicates strong strings but does not generate new ones. The crossover operator is applied to the mating pool, with the aspiration of yielding improved offspring. Traditional genetic algorithms frequently employ single-point crossover, wherein two mating chromosomes are cut once at corresponding positions, and the segments after the cuts are exchanged.
The crossover probability (Pc) is a significant parameter, dictating the frequency of crossover operations, ranging from 0% (no crossovers) to 100% (all offspring formed via crossovers) [11], [13].

2.4 Mutation

Mutation follows the crossover phase and is instrumental in averting the convergence of all solutions in the population towards a local optimum. Traditionally perceived as a simple search operator, mutation complements crossover by exploring the entirety of the search space. Mutation acts as a background operator, maintaining genetic diversity within the population.
For binary encoding, random bit flips from 1 to 0 or vice versa constitute common mutation methods. The mutation probability (Pm) plays a crucial role, in determining how frequently chromosome segments undergo mutation, varying from 0% (no mutations) to 100% (complete chromosome alteration). A vast array of mutation techniques exists within the extensive literature on genetic algorithms [11], [13], [14].

2.5 Stopping Conditions

Genetic algorithms implement diverse stopping conditions, adapted to specific requirements. These conditions include terminating the GA when a specified number of generations has evolved, ending the genetic process after a predetermined time limit has elapsed, or concluding if the maximum number of generations is reached before the specified time limit. Additionally, the process may cease if the maximum number of generations is achieved before a specified number of consecutive unchanged generations [11].

4. METHOD- MODIFIED GENETIC ALGORITHM

As we discussed the normal genetic algorithm, in the sense of modification we made some changes in the normal genetic approach. In a normal GA approach when the population is randomly selected fitness values are evaluated and the population takes part in crossover. After crossing over of strong individuals we found new best-fits individuals and again they participated in crossover till the

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

strongest population not found. If in the crossover process, the newly generated population got the same fitness-valued chromosomes despite process will continue till the lock condition. After that mutation started to change the population.

Due to this unnecessary time & and energy of the processor is wasted in the generation of a new population. We tried here to discard the population which not important for crossover means first finding out whether the population is important for our desired results or not. If that population is not important for our desired solutions then we call mutation immediately. Checking the importance of the population is performed in each state of iteration. If in any state of iteration, our algorithm finds that no longer this population generates the desired solution then perform mutation for a new one.

To achieve our goal we made changes in the GA basic approach after random selection of population. Here below we show our approach to getting important association rules through a modified genetic algorithm;

Step 1: Take a sample data set.
Step 2: Take the user-define minimum support as an input.
Step 3: Apply Apriori algorithm;
1) First generate a frequent item set
Ck: Candidate item set of size k
Lk: frequent item set of size k
L1= {frequent items};
2) for(k= 1; Lk!=∅; k++) do begin
Ck+1= candidates generated from Lk;
3) For each transaction t in the database do
Increment the count of all candidates in Ck+1in t.
4) Lk+1= candidates in Ck+1with min_support
End
Return Lk;
Step 4: Now take inputs are Lk, min supp, min confidence,
fit= min supp*min_conf, r_sup, r_conf(rules actual conf).
Step 5: Apply modified GA for optimization
BEGIN
1) Generate the initial population randomly;
2) Compute the fitness of each individual;
If (r_sup >= min_sup) && (r_conf >= min_conf)
Fitness = r_sup*r_conf;
Else
Fitness = r_sup*r_conf*fit;
End;

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

3) Compute the mean value of all fitness of individual;
4) Compute variance of all fitness of individual;
5) Compute the standard deviation of all fitness values;
6) Compute error (err = abs (mean/fit);
7) if (dev < mean/10)&&(err > 0.15*fit)
8) Call mutation;
9) if (dev > mean/10)&&(err < 0.10*fit)
10) Call mutation;
11) Else run a normal genetic algorithm
12) Select individuals from the population for mating;
13) Create offspring by applying recombination and/or mutation to the selected individuals;
14) If the population has converged
15) Than finishes: =true;
16) END
Terminate algorithm when found desired fitness valued solution or complete no of iterations and finally;
17) Result =Optimized rule generated.

Further, we discuss databases and results applied to our modified genetic algorithm. In our approach, we use the term min supp as the minimum user-defined support value, and min conf as the minimum confidence value. Desired fitness value means that fitness value which we calculate from the product of min supp and min conf. Due to our approach, we save extra iteration time and converging time. We execute our approach with normal genetic and apriori algorithms and the results are in our goal means our algorithm performs better than the others which we show in the next section. Our applied conditions remove the problem of local minima and large spreading in search space. If the mean value is less than the desired fitness value and the variance is greater than the desired fitness value. It means search space is spread and conversion towards the global solution. If the mean value is more or similar to the desired fitness value and the variance is smaller than the desired fitness value, it means space is less spread and conversion takes place so, continue the iteration because we will get the desired solution soon.

5. EXPERIMENTAL RESULTS AND DISCUSSION

Table 1. Sample item sets from Food Mart 2000 data

T.ID	ITEM SET
1	Canned Soup, Seafood, Snack Food, Pizza
2	Paper Products, Starchy Foods, Pizza, Jam
3	Paper Products, Seafood, Snack Food, Pizza
4	Starchy Foods, Paper Products, Pizza, Snack Foods

T.ID	ITEM SET
5	Canned Soup, Starchy Foods, Bread, Jam
6	Canned Soup, Starchy Foods, Pizza, Bread
7	Paper Products, Starchy Foods, Jam, Bread
8	Canned Soup, Paper Products, Bread, Pizza
9	Seafood, Paper Products, Snack Foods, Pizza

To assess the potential of our proposed approach during the experiment, we have used a Microsoft SQL inbuilt data set named FoodMart 2000. The FoodMart 2000 database is a sample database for a supermarket in Microsoft SQL Server. Here below we show a table of our sample data set of goods over which we performed Apriori, normal genetic, and our proposed approach. Table 1 clearly shows that the first transaction ID contains canned soup, seafood, snack food, and pizza items in an item set. We adopt the approach of the Apriori algorithm for generating both association rules either positive or negative. As we discussed before the Apriori algorithm generates frequent item set according to their support value and after that generate rules.

After applying Apriori algorithms we got a frequent item set. Further, we show graphical results of support and confidence values from the non-optimized rule which are evaluated by frequent item set after their presence and absence comparison to get both positive and negative rules. Figure 1 shows association rules and support values, Figure 2 shows association rules and confidence values.

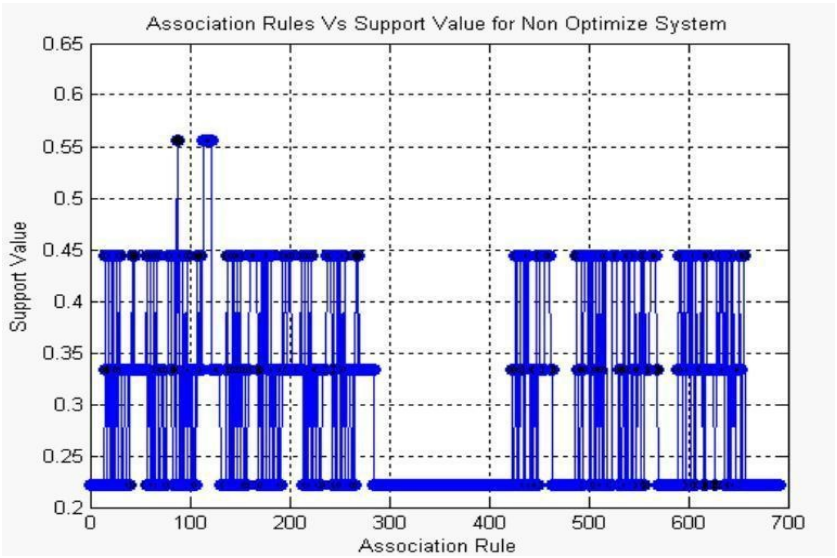

Figure 1. Support values for association rules

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

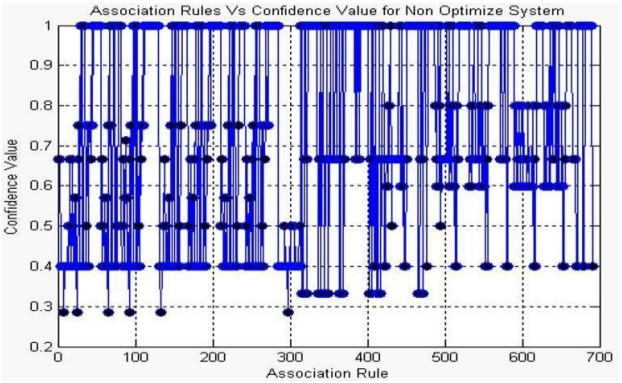

Figure 2. Confidence value for association rules

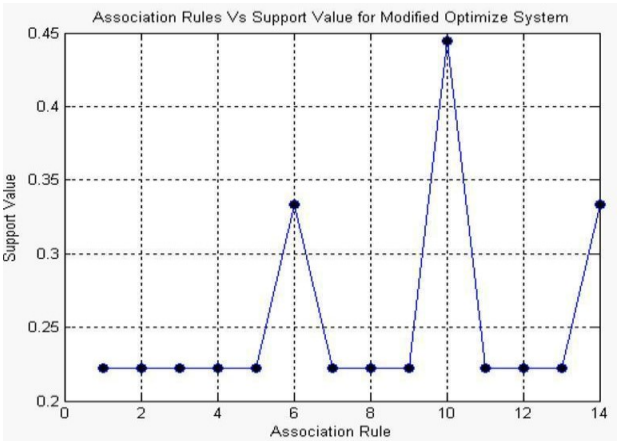

Figure 3. Optimize rules with support values

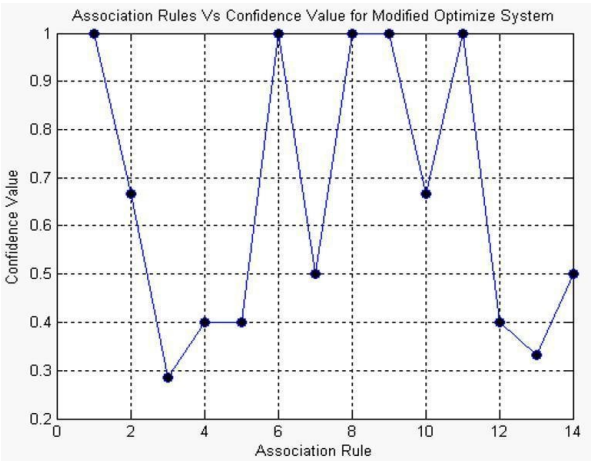

Figure 4. Optimize association rules with confidence values

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

Figure 3, Figure 4, and Figure 5 show optimized association rules with their support, confidence, and fitness values respectively. All results were generated from our modified approach. In Figure 5 we see that effective association rules with their desired fitness values. After executing no of times our approach provides more and more effective rules. In Figure 5 we easily examine the difference between a simple Genetic algorithm and a Modified Genetic Algorithm. A blue dot indicates the fitness versus association rule for the Simple Genetic algorithm and red stars indicate for modified Genetic Algorithm.

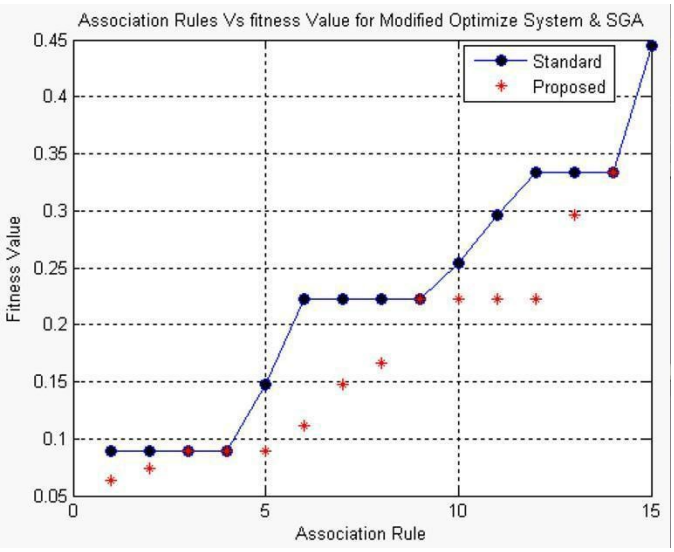

Figure 5. Optimize rules with their fitness values.

6. FUTURE SCOPE

Existing relevant studies such as [14], and [15] have focused on the combination of ARM and evolutionary algorithms to perform frequent pattern mining. Such studies lack data scalability therefore to overcome this issue in the future we will going to adopt the state-of-the-art (SOTA) artificial intelligence-based techniques in combination with evolutionary algorithms. In recent years such SOTA techniques have proved their prowess in different domains such as healthcare [17], [18], cyber security [19], and misinformation detection [20], [21]. Thus, we believe that AI-based techniques will perform well in the area of association rule mining for retail sectors. We will also try to apply other optimization algorithms for association rule mining. We also try to modify their fitness functions or create effective fitness functions to improve the efficiency of optimization. We already applied this approach to the food mart dataset and will apply it to other domains such as the healthcare database.

7. CONCLUSION

This work shows our approach to generating efficient association rules which are helpful in many application domains. In this work, we have applied our approach in the retail domain by utilizing the publicly available Food art dataset. Our approach has outperformed the base genetic algorithm. Our key finding shows that the mean and standard deviation-based fitness function of a GA has the potential to extract the optimal set of frequent item sets via the Apriori algorithm.

REFERENCES

[1] E. Ramaraj and N. Venkatesan, "Positive and negative association rule analysis in health care database," *Int. J. Comput. Sci. Netw. Secur.*, vol. 8, no. 10, pp. 325–330, 2008.

[2] M. A. Mahdi, K. M. Hosny, and I. Elhenawy, "FR-Tree: A novel rare association rule for big data problem," *Expert Syst. Appl.*, vol. 187, p. 115898, 2022.

[3] R. V. Prakash, D. Govardhan, and D. S. Sarma, "Mining frequent itemsets from large data sets using genetic algorithms," *Artif. Intell. Tech. Approaches \& Pract. Appl.*, no. 4 SPEC. ISSUE, pp. 38–43, 2011.

[4] A. Sohail, "Genetic algorithms in the fields of artificial intelligence and data sciences," *Ann. Data Sci.*, vol. 10, no. 4, pp. 1007–1018, 2023.

[5] R. Agrawal, R. Srikant, and others, "Fast algorithms for mining association rules," in *Proc. 20th int. conf. very large data bases, VLDB*, 1994, pp. 487–499.

[6] M. Saggar, A. K. Agrawal, and A. Lad, "Optimization of association rule mining using improved genetic algorithms," in *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583)*, 2004, pp. 3725–3729.

[7] C. Cornelis, P. Yan, X. Zhang, and G. Chen, "Mining positive and negative association rules from large databases," in *2006 IEEE Conference on Cybernetics and Intelligent Systems*, 2006, pp. 1–6.

[8] H. Zhu and Z. Xu, "An effective algorithm for mining positive and negative association rules," in *2008 International Conference on Computer Science and Software Engineering*, 2008, pp. 455–458.

[9] R. Dewang and J. Agarwal, "A new method for generating all positive and negative association rules," *Int. J. Comput. Sci. Eng.*, vol. 3, no. 4, pp. 1649–1657, 2011.

[10] S. Das and B. Saha, "Data quality mining using genetic algorithm," *Int. J. Comput. Sci. Secur.*, vol. 3, no. 2, pp. 105–112, 2009.

[11] S. N. Deepa and S. N. Sivanandam, "Principles of soft computing." Delhi, India: Wiley India Pvt. Ltd, 2011.

[12] M. Anandhavalli, S. K. Sudhanshu, A. Kumar, and M. K. Ghose, "Optimized association rule mining using genetic algorithm," *Adv. Inf. Mining, ISSN*, vol. 9753265, 2009.

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

[13] S. Ghosh, S. Biswas, D. Sarkar, and P. P. Sarkar, "Mining frequent itemsets using genetic algorithm," *arXiv Prepr. arXiv1011.0328*, 2010.

[14] P. Bajpai and M. Kumar, "Genetic algorithm--an approach to solve global optimization problems," *Indian J. Comput. Sci. Eng.*, vol. 1, no. 3, pp. 199–206, 2010.

[15] P. Vyas and J. Dubey, "An Efficient Methodological Study for Optimization of Negative Association Rule Mining," *IJCA, ICRTITCS*, vol. 3, pp. 27–31, 2013.

[16] P. Vyas and A. Chauhan, "Comparative optimization of efficient association rule mining through PSO and GA," *Proc. - 2013 Int. Conf. Mach. Intell. Res. Adv. ICMIRA 2013*, pp. 258–263, Oct. 2014, doi: 10.1109/ICMIRA.2013.55.

[17] P. Vyas, K. N. M. Ragothaman, A. Chauhan, and B. P. Rimal, "Classification of COVID-19 Cases: The Customized Deep Convolutional Neural Network and Transfer Learning Approach," *AMCIS 2022 Proc.*, Aug. 2022, Accessed: Dec. 08, 2022.

[18] P. Vyas, K. Ragothaman, A. Chauhan, and B. Rimal, "Classification of COVID-19 Cases: An Exploratory Study by Incorporating Transfer Learning with Cloud," *MWAIS 2021 Proc.*, May 2021.

[19] P. Vyas, G. Vyas, and A. Chennamaneni, "Detection of Malicious Bots on Twitter through BERT Embeddings-based Technique," in *AMCIS 2023 Proceedings*, p. 6, 2023.

[21] P. Vyas, J. Liu, and O. El-Gayar, "Fake News Detection on the Web: An LSTM-based Approach," *AMCIS 2021 Proc.*, Aug. 2021.