ELITE VOTING CROSSOVER AND MUTATION OF GENETIC ALGORITHM FOR FEATURE SELECTION

Bakari, K. J.

Department of Mathematical Sciences, Faculty of Science, Taraba State University, Jalingo, Taraba State Nigeria.
Email: kamalbakari@gmail.com

ABSTRACT

In record classification, not all available attributes might be both useful and relevant. Hence the need f feature selection arises. Over the years, the Machine Learning community has used a number of algorith for feature selection. One of such algorithms is Genetic Algortihm (GA). Recently, these algorithms have se a fall in performance due to growth in data dimensionality and sample size. This calls for efforts to enhan the performance of this algorithm to meet the current trends. Given that the performance of GA hea depends on the parameter values used and genetic operators applied, it is pertinent to improve t performance of GA by improving the genetic operators. To this end, this work introduced new gene operators (crossover and mutation) of the GA to be used in Feature Selection (FS).The efficiency of proposed algorithm was measured using Extreme Learning Machine (ELM) with Cleveland dataset University of California Irvine Machine Learning repository. The result shows a promising improvemen performance. The proposed algorithm is particularly important in situation of time constraints (online F and low computation power availability.

Keywords: *Feature Selection, Genetic Algorithm, Mutation, Crossover, Voting*

INTRODUCTION

With the growth in both data dimensionality and sample size due to the emergence of fields such as bioinformatics (Xianlong *et al.*, 2013), "the omics" (Tong *et al.*, 2015) and the falling cost of data capture and storage; FS has proven to be an indispensable procedure in machine learning. Furthermore, data is collected for various purposes and for use with different algorithms. These data might end up been used in different processes and with different algorithms, hence the need to select features which are compatible with the target algorithms. In essence the two major problems associated with redundant and irrelevant features are:

- Unnecessary computational cost
- Overfitting

Thus to prevent these, only useful and relevant features must be used in ML. the task of FS is: given a dataset with n attributes, to find a set of m distinct attributes which provide best performance of the learning algorithm. This can be mathematica expressed as;

$$\min z = \left(\frac{n!}{(m-n)!m!}, -f(w)\right)$$

S.T. $x_i \leq n, \qquad x_i \in X$

Where $n!$ & $m!$ is the factorial of n and respectively, $f(w)$ is the measure of goodness of t classification algorithm.

For binary classification problem, ${}^{n}C_{m}$ can b simplified to 2^n. Where n is the total number attributes in the dataset. Therefore for a dataset wi 100 attributes, there are 2^{100} or 1.2677 10^{30}different combinations. This makes t application of metaheuristic search inherent. This because

1. They are less computationally expensive.
2. Only search promising areas of the sp there by reduce complexity.

Therefore, this work intends to further enhance performance of GA which is a metaheuristic to

used for FS by introducing a special crossover and mutation to the GA.

Record Classification is the task of categorizing records into classes based on some known (labeled) training dataset (Dash and Liu, 1997). There are different algorithm and models used for classification some of which are statistical (Haleh and Kenneth, 2012) neural network (Ma and Huag 2008), kernel based classifier (Tong *et al.*, 2015), decision trees Bir and Yinqiang 2003), ensemble models (Xiaolong *et al.*, 2013) etc. The process of classification is divided into

- **Training phase-** dataset is classified into categories based on attributes values and class conditional probabilities of the records in the training data. In this phase, the learning algorithm learns a mapping from the attribute space to the class space as follows:

$$f(attributes) \rightarrow class$$

In the process of learning the induction algorithm tends to optimize the parameters as best as possible on the given data this leads to overfiting. This is more probable in a case where the data observations are few and the attributes (include redundant and irrelevant attributes) are much or where the model parameters are many (Muthanantha and Ramakrishnan, 2004). For example in a medical records classification where the patient ID is included as an attribute, an over tuned learning algorithm may conclude that the medical condition been modeled is associated with the patient ID (Dash and Liu, 1997). This is the main reason why all irrelevant and redundant attributes must be discarded form the dataset before starting the process of training. There are different approaches used for feature selection which were earlier discussed in this work.

Another problem associated with the training phase is the computational complexity and convergence of learning algorithm. Learning algorithms such as GD (Dash and Liu, 1997) which use the back propagation training methods take time to train and to converge this is because the value of parameters are tuned one at time which requires more time to achieve. To overcome this, Guang-Bin Huang proposed the ELM which is a SLFN with minimal parameter tuning for faster training time. This is because the weight connecting the input nodes and the hidden layer are set once and never updated. A model of ELM can be expressed as

$$\check{T} = \beta H \text{ where } H = h(w_k . x_i)$$

Where h is some activation function (usually sigmoid, radial basis, Gaussian, Logistic or any other binary or bi-polar function for classification or linear function in the case of regression) W *is* vector of weights connecting the input and hidden layer and W_2 is the vector of weights connecting hidden layer and the output layer. In a dataset of N observations with samples $\{x_i, y_i\}$ is assumed. Where $x_i \in R^d$ and $y_i \in \{1, 2, \ldots, c\}$ and c is the total number of different classes. A binary variable is used to encode the target T. Here, T is the vector of class tags where $T_{ij}=1$ if and only if $y_i = j$ (that is the instance 1 is a member of class j) Else $T_{ij}=0$. In the case of a bi-class classification problem, a single output variable is enough since membership to a class can be expressed using a threshold. Therefore

$$\hat{Y} = f(x) = \sum_{k=1}^{M} \beta_k h(w_k . x_k)$$

From an algebraic point of view, the problem is that of calculating the least square of the following matrix equation above. The model bias may be represented by concatenating a 1 to each x_i appending a column of 1's to the matrix H. Therefore, the activation can be further written as

$$\sum_{i=1}^{\tilde{N}} \beta_i h_i(w_i . x_j + b_i) = 0_j$$

Where b_i is the threshold of the i^{th} hidden node. Hence, $w_i . x_i$ represent the inner product of w_i and x_j this can be expanded as

$$= \begin{matrix} H(w_i \dots w_{\tilde{N}}, b_i \dots b_{\tilde{N}}, x_i \dots x_{\tilde{N}}) \\ \begin{bmatrix} h(w_1.x_i + b_1) \dots . h(w_1.x_i + b_1) \\ \vdots \\ h(w_1.x_{\tilde{N}} + b_1) \dots . h(w_{\tilde{N}}.x_{\tilde{N}} + b_{\tilde{N}}) \end{bmatrix}_{N \times \tilde{N}} \end{matrix} \beta$$

$$= \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_{\tilde{N}}^T \end{bmatrix}_{\tilde{N} \times M} \quad and \; T = \begin{bmatrix} t_1^T \\ \vdots \\ t_{\tilde{N}}^T \end{bmatrix}_{\tilde{N} \times M}$$

- **The prediction phase-** in this phase the mapping function or transformation learned in the training phase is used to categorize new data instances into the established classes based on the attribute values of the new data instances. For prediction to be successful, the features and class distribution of both the training and prediction data samples must be the same.

ELM was used in (Tong *et al.*, 2015) for the identification of Erythematic Squamous Skin disease. Due to the close clinical features of the diseases, other classification algorithms perform poorly on this problem. In contrast, ELM performed astoundingly better with an accuracy of 84.74% as opposed to other methods such as the classical ANN which reported an accuracy of 77.26% on average based on the UCI Erythematic dataset. However, the study noted that ELM performs slightly better when the percentage of training sets decreases with an increase in the testing sets which is a rare ability.

Banupriya and Karpagavalli, 2015 used Electrocardiography (ECG) signal to detect cardiac disease using ELM classifier. The study compared the performance of ELM and Relevance Vector Machine (RVM) on MIT-BIH dataset. The result showed the superiority of the RVM on unprocessed dataset and vice versa on a processed. Finally, the research suggested the use of ELM in a situation where (1) data preprocessing can be performed, (2) where speed is of higher importance and learned model need to be understandable. While RVM is better applied in a situation where; preprocessing cannot be performed, speed is of low or no importance in the learning process and model comprehensibility is of low or no importance.

Muthanantha and Ramakrishnan, 2014 proposed Optimized Extreme Learning Machine (OELM) for the classification of Encephalogram (EEG) w emphasis on epileptic seizure detection. The stu showed that the speed and accuracy of conventional ELM was enhanced by ranking neurons using Minimum Redundancy Maxim Relevance (MRMR) algorithm and selection of most relevant neurons thereby reducing the size the ELM and the computational requirement of model in general. This reduction there by boo speed and accuracy of the learned model. The stu observed that ELM needs more hidden nodes th Backpropagation methods but much less than SV Furthermore, ELM models tend to have proble when irrelevant (uncorrelated) and redund variables are present in the training set.

(Tong *et al.*, 2015; Haleh, 2012) Used a distribut ELM with Mapreduce framework which can co the shortage of the conventional ELM who learning ability in huge dataset is weak. The stu found out that most expensive computational part the ELM is the Matrix Moore-Penrose generaliz inverse operator in the output weight vec calculation. Due to generalization performanc rapid training speed and little parameter tunin ELM has demonstrated it is indispensable in th area of online or real time classification of hug dataset. However, the study noted that the propos method can only be applied in a situation where th matrix multiplication is decomposable and th corresponding output weight vector can be centrall computed.

Xiaolong *et al.*, 2013 used ELM for a study on th diagnosis Attention-Deficit/Hyperactivity Disord (ADHD). They studied the effects of data volum on the performance of both SVM and ELM on th classification of which parts of the brain ar associated with ADHD. The result showed th ELM achieved 90.18% accuracy compared to SV which achieved 86.55%. The research noted th speed and accuracy of the ELM algorithm ar minimal interference requirement. In conclusion, th research proposed the use ELM in real-li examination of ADHD.

The remaining part of this work is structures follows: section 2 introduce the proposed algorith section 3 gives the experimental setup, results an discussion, Finally section 4 gives conclusion an recommendation.

Proposed Crossover and Mutation

As in the conventional GA, the proposed method begins by creating a population of randomly generated individuals. Then these individuals are evaluated using a fitness, after the normal elitism, crossover and mutation; a special process of parent selection which uses the elite individuals is performed to select individuals into the mating pool. Then, the special crossover and mutation are then applied to these individuals to generate new offsprings.

To ensure we retain the randomness of the GA, the special mutation generates a little number of offspring of the next generation. Furthermore, crossover and mutation are only applied to elite individuals to encourage greediness of the algorithm. In addition, the minimum requirement to serve as parent is averaged over the whole population. This procedure is suitable for FS because we are only interested in alleles which have higher relationship with target class. Therefore, only alleles agreed upon by elite individuals are considered as important.

Generating a New Population

A new population of chromosomes at every iteration is generated by three different recombination and alteration operations. These are

- **Conventional Elitism** – This is the process by which individual chromosomes are sent to the next generation without been altered. Usually, a percentage of the best individuals or individuals that meet some criteria are sent to the next generation in order to preserve good traits over generations hence, the name elitism. In the proposed algorithm, N' (a parameter pass to the algorithm or the conventional elitism rate might be used) individuals are selected as elite individuals to the next generation.

- **Conventional Crossover and Mutation** – next, the conventional crossover and mutation which is the process of individuals generation through recombination and alteration is used to generate N'' number of individuals in the next generation. Hence, a good crossover and mutation operations are required for a better performance.

- **Special Crossover and Mutation** – The remaining (N''') individuals in the population are generated using the proposed crossover and mutation operations which will is discussed in the next section. This operation produces fitter individuals because the elite individuals and voting mechanism are used to generate individuals at the same time average population fitness is used for performance measurement to prevent premature convergence. Therefore, the composition of the next generation at any iteration can be diagrammatically represented as in figure 1:

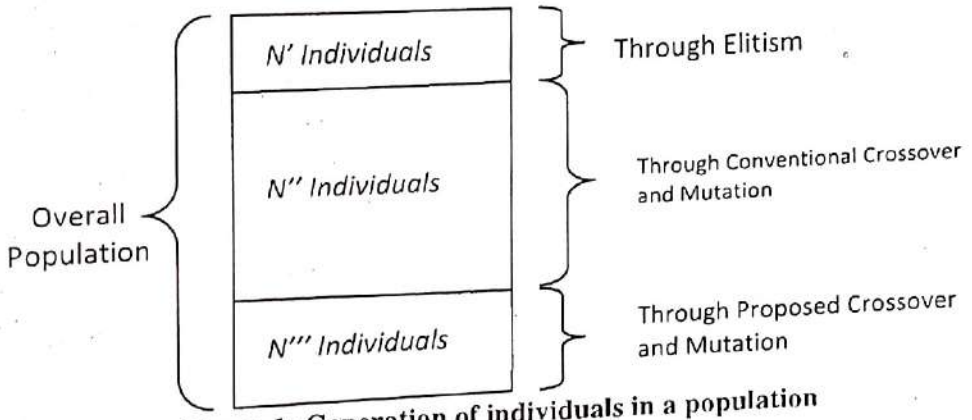


Figure 1: Generation of individuals in a population

Formulation of the New Individual

In a GA with a population of N individuals where each individual is composed of M alleles, then this population can be represented as a matrix of $N \times M$ dimension. The fitness of each individual is denoted by $f(indiv_n)$, the fitness required to be considered as a good parent is denoted as f_g, the average fitness in the population is denoted by F_{avg}. $indiv_n : f(indiv_n) \geq f_g$ are selected to undergo the special crossover and mutation to create a new individual. That is to say out of N individuals, N' good individuals (those with fitness $\geq f_g$) are selected for the proposed reproduction. The value of f_g is obtained using

$$f_g = g \times F_{avg} \quad (3.1)$$

Where g is a constant [0, 1] which signifies the relevance of the average fitness in the process and F_{avg} is the average fitness in the population and is given by

$$F_{avg} = \frac{\sum_1^N f(indiv_n)}{n} \quad (3.2)$$

If $g=1$ then, f_g will be equal to F_{avg}. The reason for selecting g [0, 1] is to ensure that the whole search space is been explored. After obtaining the minimum requirement to be selected as a parent (i.e. F_{avg} and f_g), the sum of 1's alleles across both horizontal and vertical directions of the matrix ($N \times M$) is obtained. The sum of alleles in the horizontal direction serves as indicator of the number of alleles which should be present in the new individual and is obtained as

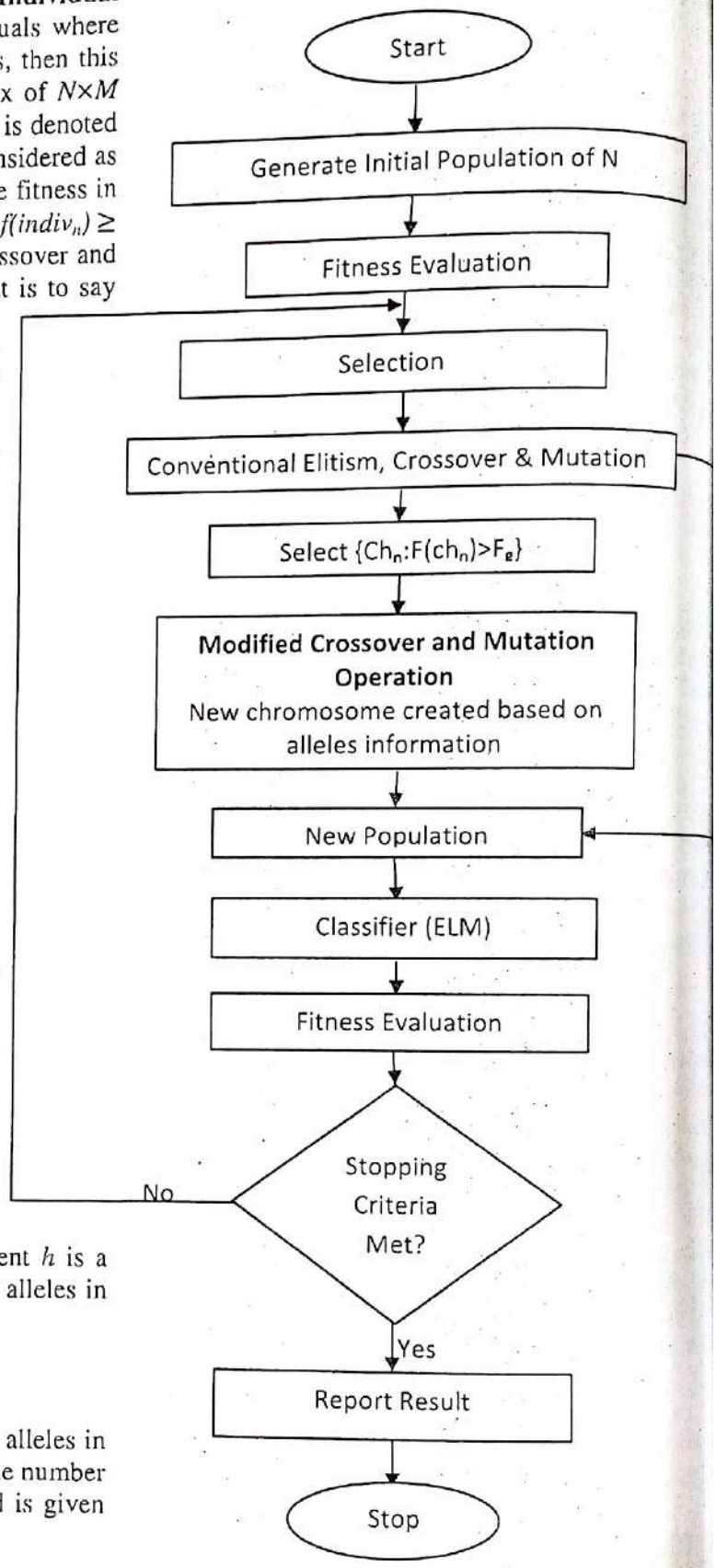
$$L = h \times L_{avg} \quad (3.3)$$

Where L is the number of 1's in the parent h is a constant [0, 1] and L_{avg} is the average 1's alleles in the horizontal direction and is given as

$$L_{avg} = \frac{\sum_{n=1}^{N'} L_n}{N} \quad (3.4)$$

Where L_n is the sum of occurrences of 1's alleles in the horizontal direction which represents the number of attributes selected by an individual and is given by

$$L_n = \sum_{m=1}^{M} a_{nm} \quad (3.5)$$



And the sum of 1's in the vertical direction is the voting weight of a selected feature that determines which allele should be a 1 in the generated offspring and defined by

$$V_m = \sum_{m=1}^{M} a_{nm} \quad (3.6)$$

The created offspring will be composed of 1 alleles selected from the highest constant V_m $m=1$ *to* M. A single individual is considered for mutation using bit flip mutation where a single allele with a b_i value of one standard deviation below the mean (i.e. 1 value below L_{avg} is flipped from a zero to a 1 to generate another individual. More individuals are generated by repeating this procedure for all other alleles with one value below L_{avg} until the required number of N''' is obtained

In the example above, L_{avg} =7 and $l = L_{avg}$. Hence, the new individual will have 7 1's alleles. For us to get these alleles we use the biggest vertical alleles which are found at m=*13, 12, 6, 10, 9, 2, 3* with values 9,8, 8, 7, 7, 7, 6 respectively, these alleles are then ranked based on V_m and the top l ranked alleles are chosen. Here, we chose 6 which is the next rank allele because 7 cannot be chosen.

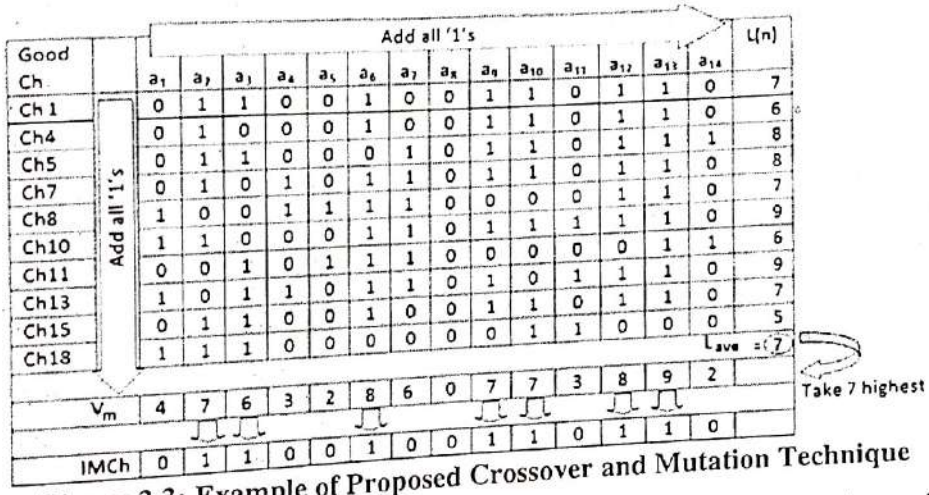


Figure 3.3: Example of Proposed Crossover and Mutation Technique

RESULTS AND DISCUSSION

In this section the results obtained from the computational experiment are presented. Here, Genetic Algorithm with the proposed crossover and mutation methods was used against the selected dataset and the performance was compared with the performance of the conventional algorithm.

Furthermore, the GA used two different fitness functions in this research (1) a fitness function which penalizes individual chromosomes which select more features as proposed by (Dash and Liu, 1997). (2) A fitness function which uses only the classifier accuracy. The performance was evaluated and finally, the proposed GA operator was added to the GA and the performance was evaluated. The evaluation metrics used in this work included (1) Accuracy of the classifier (2) Convergence of the algorithm (3) Average individual fitness (4) Diversity of individuals (5) Number of features selected by best individual.

Other investigated issues include effect of parent pool size on the proposed algorithm, effect of population size on the convergence of the algorithm and stability of the GA using the proposed algorithm, the effect of probability of mutation and crossover on the proposed method. The experiment started with the conventional algorithm then a special fitness function was added to the GA. Thereafter, the proposed crossover and mutation was added with and without the fitness function. Here, an ELM with 10 hidden neurons was trained to access the performance of the proposed method.

This study started by investigating the performance of the conventional GA on the dataset. The GA used a population size of 50, maximum iteration of 100, elitism of 25%, and crossover rate of 20% and mutation rate of 2% as suggested by (Tong *et al.*, 2015; Haleh, 2012). The number of bits is equal to the total number of features in the dataset. Thus at the end of an experiment, the number of alleles with a 1 signifies those features which are selected for example a chromosome with the following alleles 0011101 on the dataset means the 3rd,4th,5th and 7th attributes were selected. The result of conventional GA is presented in the figure 1:

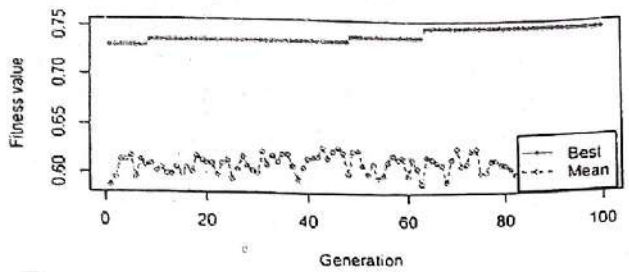


Figure 1: Result for Conventional GA without special crossover and mutation, without special fitness function.

In the Figure 1 above, the convergence of the conventional GA on the test datasets is presented. Here, the fitness function did not penalized individuals with higher number of selected features. It can be seen that the algorithm achieved an accuracy of 0.76 which is below the average reported accuracies (Haleh and Kenneth, 2012; Bir and Yinqiang, 2003; Wu et al., 2010) and the algorithm converged at the 64th iteration. Furthermore, the best fitness achieved here is 0.76 while the average fitness is 0.54 which points to the wide gap between individual fitness using this algorithm. Therefore, in situations where multiple equally fit individuals are required this algorithm may not be of any use. Again the fact that the algorithm improved even at the 64th iteration can be seen as yet to converge because the achieved accuracy is below the baseline reported accuracies as earlier noted

Next the GA was integrated with a special fitness the function which penalizes an individual for selecting more features as discussed above. Theoretically, this is expected to force all individuals to conform to some environmental factors thereby making them more fit and so the population will be highly qualitative. The result of this experiment is presented below:

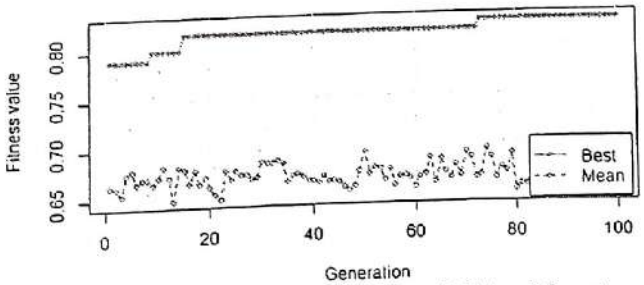


Figure 2: Result for Conventional GA without special crossover and mutation with special fitness function.

In the Figure 2 above, all the other algorit parameters and settings were kept same as in t first experiment only the special fitness funct was introduced. The result indicates a relat improvement in the convergence time but achieved accuracy remained the same. At the end this experiment the algorithm selected 4 34 featur out of the original 76 features as the best indicato in the dataset. This indicates that the algorithm w this parameters and settings performed below reported baseline performance (Haleh and Kenne 2012; Bir and Yinqiang, 2003; Wu et al., 20 Geetha and Geethalakshmi, 2011).

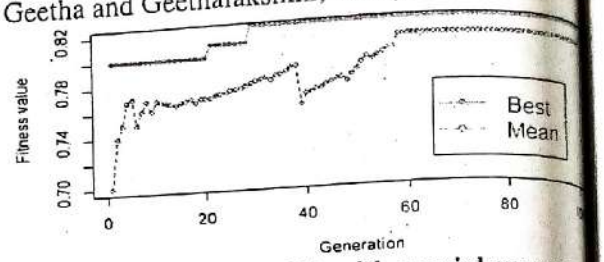


Figure 3: Result for GA with special crossover and mutation without Special fitness function

In Figure 3 above, the proposed crossover a mutation were introduced to the GA. As in the tw previous experiments, all other parameters a settings were unchanged. Furthermore, a simp fitness function which uses only the classifi accuracy (does not penalize individuals with highe selected features) was used. The result showed leap in achieved accuracy of the classifier from 0.7 in the experiment two above to 0.8 Proportionately, the algorithm converged at the 25 iteration which is an improvement over the tw earlier reported experiments. Furthermore, it can b seen that the difference between the best a average fitness in the population over ha stabilized over the after the convergence of t algorithm for all the datasets:

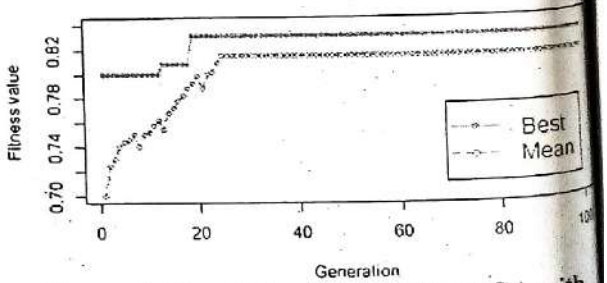


Figure 4: Result for Conventional GA with special crossover and mutation and fitness function.

In the Figure 4 above, the proposed crossover a mutation was used in addition to a fitness functi

which uses both classifier accuracy and higher selected feature penalty. All other algorithm parameters and settings were kept constant. This is to maintain consistency in the test setup for the datasets. It can be seen that the difference between the best and average fitness was relatively low in the dataset. Furthermore, the achieved classifier accuracy improved tremendously from what was reported in all the other experimental setups. More so, the number of selected features is also less than the 13 reported by most studies (Haleh and Kenneth, 2012; Bir and Yinqiang, 2003; Wu *et al.*, 2010; Xu *et al.*, 2009). Finally, it is worth noting that the average individual fitness in this setup is almost close to the best fitness which is an added advantage of the algorithm. This is useful in situations where more than one solution is required. One of the parameter settings of a GA is elitism size. This parameter determines which individuals go to the next generation without been changed thereby, preserving good traits in the population. This parameter was also used as a determinant of the parent pool size in the proposed algorithm, therefore, this work investigated the effect of elitism on the performance of the proposed algorithm and the result is presented in the table below:

Elitism	No Iteration Before Convergence	Classifier Accuracy	No of Features Selected
0	65	0.751	31
10	19	0.81	10
20	27	0.0.792	11
30	51	0.79	13
40	24	0.0.78	18
50	32	0.765	21
60	62	0.76	23
70	67	0.756	25

Table 1: Result of algorithm convergence for different elitism size on the proposed algorithm

In the Table 1 above, it can be seen that as the size of elitism increased the convergence performance of the algorithm, classifier accuracy dropped and the number of selected features increased. This is as expectedly due to the fact that as the parent pool size increase the sum of vertical important alleles will increase thereby making much allele important. Therefore, the algorithm will be unable to drop as much unimportant alleles as possible and will require more iterations to converge. It is expected that using a very high rate of mutation will solve this problem as mutation will introduce new information into the population. This is subject to further research.

Another parameter of GA which affects the behavior of the algorithm is population size. This parameter represents the number of solutions generated in each iteration and thus a variation in the population size will determine how long it takes to get the best individual. Hence, this work investigated the effect of population size on the performance of the proposed algorithm and result is presented in the table below:

Population Size	No Iteration Before Convergence	Classifier Accuracy	No of Features Selected
20	62	0.74	24
25	58	0.76	21
30	52	0.765	21
35	44	0.78	19
40	32	0.786	17
45	21	0.792	16
50	18	0.818	12

Table 2: Result of algorithm convergence for different population size on the proposed algorithm

In Table 2 above, it can be seen that as the size of the population increased the convergence performance of the algorithm improved. This is because at each iteration the number of individuals created or recombined is high. Therefore, the probability of obtaining the best individual is increased as the number of individuals in an iteration increased.

In GA, the crossover rate is the probability of an individual to undergo the recombination process. This parameter controls the intensity of the search process in the GA. Been one of the proposed techniques in this work, we investigated how different values of the parameter affects the learning process. Whence, different values of crossover rate where set for the GA using the same parameters as in the other experiments. The result is presented below:

Crossover Rate	No Iteration Before Convergence	Classifier Accuracy	No of Features Selected
40	43	0.77	19
45	39	0.79	19
50	26	0.793	18

55	22	0.798	17
60	20	0.80	15
65	16	0.804	13
70	14	0.81	10

Table 3: Result of algorithm convergence for different probability of cross over on the proposed algorithm

From Table 3 above, it can be seen that the number of iteration before convergence, classifier accuracy and number of selected features selected improved slowly as the crossover rate increased. Thus, as the intensity of the search increased, the required number of iteration reduced and more number of good individuals were generated which improved the classifier performance.

Mutation rate is the probability that an individual will undergo asexual reproduction i.e. new individuals will be generated from it without combining its traits with those of other individuals. This work investigated the effect of mutation rate on the proposed technique and the result obtained is presented below:

Mutation Rate	No Iteration Before Convergenc	Classifier Accurac	No of Features
0.4	62	0.78	22
0.5	51	0.78	21
0.6	48	0.79	19
0.7	44	0.81	16
0.8	35	0.812	14
0.9	28	0.82	12
1	22	0.83	12

Table 4: Result of algorithm convergence for different probability of mutation on the proposed algorithm

In Table 4, the mutation rate had more impact on the number of iteration before convergence but had little impact on the classifier performance. This can be from the fact that as mutation is increased the GA search diversity increases and the search tends to behave more like a random search. Thereby, requiring more iterations to converge but not affecting its ability to find better individuals faster.

In this study, the proposed algorithm was repeated for 10 times with the dataset and the convergence iteration of the algorithm in each of the experiments was compared with that of the conventional GA. The result is presented in the Figure 5 below:

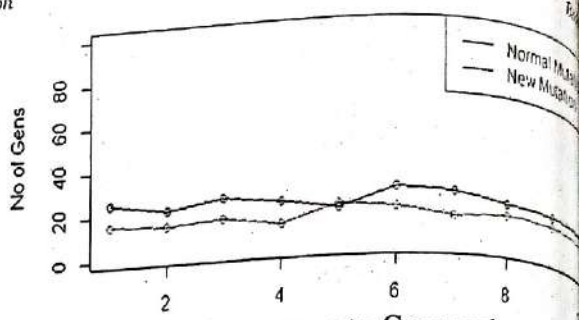


Figure 5: Convergence Comparison of Conventional and Modified GA

It can be seen from the table above the propo algorithm has consistently converged before the 2 iteration. While, by comparison the convention algorithm has always converged after the 2 iteration. In essence, the worst performance of proposed method is the same as the b performance of the conventional algorithm in all experiments.

CONCLUSION

This work reviewed the problem of d dimensionality growth in ML in current time Furthermore, it reviewed the current approach used for the task of feature subset selection in situation where speed and computational cost are high importance.

Additionally, issues that affect the performance the proposed method such as probability crossover and mutation, elitism and population si were also investigated. The result obtained show that the proposed algorithm performed better wi elitism and crossover rate were high and populati size and mutation rate are low.

RECOMMENDATIONS

Finally, it is worth noting that in a situation whe all individuals in the mating pool give the sa importance to all features in the dataset (i.e. equ vote for all alleles) this algorithm may not applicable. Based on the received results from t study, an investigation of the proposed method premature convergence is highly recommend This is because of the seeming fast convergence the algorithm

REFERENCES

Haleh Vafaie, and Kenneth De Jong (201 *Genetic Algorithms as a Tool for Feat Selection in Machine Learning*, Springer.

Tong Liu, Liang Hu, Chao Ma, Zhi-Yan Wang, Hui-Ling Chen (2015). A Fast Approach

Detection of Erythemato-Squamous Diseases Based On Extreme Learning Machine With Maximum Relevance Minimum Redundancy Feature Selection *International Journal of Systems Science* Volume 46, Issue 5.

Xiaolong Peng, Pan Lin , Tongsheng Zhang, and Jue Wang (2013). Extreme Learning Machine-Based Classification of ADHD Using Brain Structural MRI Data Published: November 19, 2013.

Dash M. and Liu H. (1997). Feature selection for classification. Intelligent Data Analysis.

Ma S. and Huang J. (2008). Penalized Feature Selection and Classification in Bioinformatics. *Briefings in Bioinformatics*, 9(5):392–403.

Muthanantha Murugavel A. S., and Ramakrishnan S. (2014). An Optimized Extreme Learning Machine for Epileptic Seizure Detection *International Journal of Computer Science*, 41:4, 30 November 2014.

Bir Bhanu, and Yingqiang Lin (2003). Genetic Algorithm Based Feature Selection for Target Detection in SAR Images, *Image and Vision Computing Journal* 21 (2003) 591–608

Banupriya C.V., and Karpagavalli S. (2015). Electrocardiogram Beat Classification Using Support Vector Machine and Extreme Learning Machine ICT and Critical Infrastructure: Proceedings of the 48th Annual Convention of Computer Society of India- Vol I Volume 248 of The Series Advances in Intelligent Systems and Computing pp 187-193

Wettschereck, D. Aha D. and Mohri T. (1997). A Review and Empirical Evaluation of Feature Weighting Methods for a Class of Lazy Learning Algorithms. *Artificial Intelligence Review*, 11:273–314, 1997.

Wu, X. Yu, K. Wang, H. and Ding, W. (2010). Online Streaming Feature Selection. In Proceedings of the 27th International Conference on Machine Learning, pages 1159–1166, 2010.

Xu, Z. Jin, R. Ye, J., Lyu, M. and King, I. (2009). Discriminative Semi-supervised Feature Selection via Manifold Regularization. In IJCAI' 09: Proceedings of the 21th International Joint Conference on Artificial Intelligence, 2009

Geetha, G. and Geethalakshmi S. N. (2011). Detecting Epileptic Seizures Using Electroencephalogram: A New and Optimized Method for Seizure Classification Using Hybrid Extreme Learning Machine International