İstanbul
Bilgi Üniversitesi

Feature Selection Using Genetic Algoritm

by

Ece Akçıçek, 117200059

Supervised by

Tuğba Dalyan

Submitted to the
Faculty of Engineering and Natural Sciences
in partial fulfillment of the requirements for the

Bachelor of Science

in the
Department of Computer Engineering

January, 2021

Abstract

Simplicity is good. Explaining and interpreting a model with too many variables can become quite complex. With feature selection, the overfitting situation that may occur in the model can be prevented. Data sets that contain large number of attributes can lower the efficiency of the system and damage time complexity. Feature Selection is an important preprocessing step to avoid computational cost, and increase efficiency. In a model with a lot of features, the training performance can be extremely good. However when tested, if the situation is reversed, there may be features that cause overfitting somewhere. If these features are eliminated, true accuracy of the model can be increased. Genetic Algorithm contains mutation and cross-over phases, which makes GA a strong method for feature selection. Purpose of this paper is to implement a Genetic Algorithm that provides optimum fitness, by keeping the relevance of the features to label at the maximum level and keeping the number of features at a minimum level.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

GA	Genetic Algorithm
ML	Machine Learning
FS	Feature Selection
GAW	Genetic Algorithm Wrapper
PCA	Principal Component Analysis
SVM	Support Vector Machine
KNN	K-Nearest Neighbors
MIM	Mutual Information Maximization
AGA	Adaptive Genetic Algorithm
TS	Tabu Search
SFS	Sequential Forward Selection
HFS	Hierarchical Feature Selection
GWO	Grey Wolf Optimization
PSO	Particle Swarm Optimization
MAD	Mean Absolute Difference

1 Introduction

For a system to be successful, there are some important measurements. Among others, some can be named as time complexity, computational cost and accuracy rate. Even if the success/accuracy rate is high, time complexity and computational cost problems can decrease the efficiency of the system.

Datasets with high dimensions can harm the efficiency by increasing complexity and computational cost. This datasets can also cause overfitting and "curse of dimensionality". To avoid/minimize these problems, Feature Selection can be applied to eliminate features that are redundant and/or non-informative.

Feature Selection is a preprocessing stage for data to derive features from feature set to reduce costs, increase classifier accuracy and efficiency.

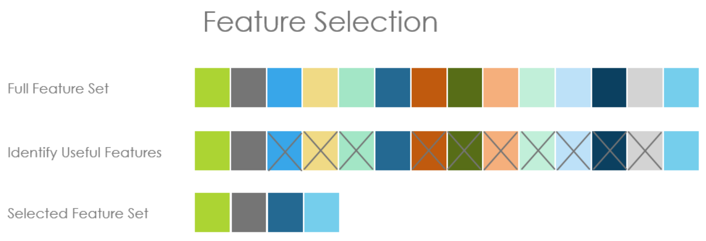


Figure 1: Feature Selection

Redundant and non-informative are two distinct properties of a feature. A relevant feature may be redundant in the presence of another feature, which is relevant, or vice versa. This strong correlation between fundamentally relevant features creates the need of a strong feature selection algorithm.

An intuitive approach to the problem may result in better selection and can lower the computational effort. This intuitiveness that is called metaheuristic approach does not guarantee to provide a global optimal solution, but provides practical efficacy. On large datasets, metaheuristic methods can iterate faster as well.

Aim of this project is to offer a metaheuristic approach to feature selection, to increase the efficiency. Genetic Algorithm will be used for feature selection in this paper.

2 Related Work

Many different approaches on feature selection have been proposed in literature, and some are shown here. Some of these base around basic GA implementation for feature selection, while some are combined with optimizers or other classifiers, aiming to derive better outcomes.

- Study [Raymer, Punch, Goodman, Kuhn, Jain, (2000)] [1] works with an approach that GA feature extraction in selection and classifier training is performed synchronously. With this approach, features are measured by their relevancy to classifying, and a weight is assigned. Genetic algorithm optimizes the vector of feature weights, depending on their relevancy, through every iteration. For classification, k-nearest neighbors method is used and results are compared with classical FS and FE methods. In testing GA Feature Extractor is shown better results than other classical methods such as Bayes, Neural Network etc.

- In [Jadhav, He, Jenkins, (2018)] [4], feature selection is used in credit scoring applications. GA wrapper (GAW) algorithm is used. Their approach follows a similar route as before, based on ranking the features by their relevance.The ranking directly effects the contribution of the features towards classification. Since wrapper approach for feature selection is a costly method, Information Gain is used to guide the feature selection. Then, propagation is made on top m features through the GAW algorithm.

  This paper uses three different classical ML algorithms such as K-nearest neighbors, Naive Bayes and Support Vector Machine (SVM). Algorithm is tested on three datasets with 14-24 features and +500 samples. The average prediction performance by IGDFS, Genetic Algorithm Wrapper and Baseline models are compared. Among the three machine learning algorithms investigated, accuracies for the SVM with baseline, GAW and IGDFS are consistently higher across all the datasets compared with those for KNN and NB. GAW+KNN and IGDFS+KNN have shown very little improvement in the accuracy of classification on the selected feature sets for all the datasets, compared with the baseline KNN on the full features.

- In [Lu et al., (2017)] [5], a hybrid feature selection algorithm is used, combining Mutual Information Maximization (MIM) and Adaptive Genetic Algorithm (AGA), addressed as MIMAGA. Mutual information refers to the dependent information of one random sample (x) on the other random sample (y).

The purpose of Mutual Information Maximization is to find genes that have strong dependency to all other genes in the same class. In the standard GA, cross-over probability and mutation probability are predefined variables which are fixed in the GA search process. In AGA, GA adjusts these probabilities during the searching process. With the tests held by 6 datasets, compared with three existing feature selection algorithms (ReliefF, sequential forward selection (SFS), MIM), classification rates for all classifiers are higher than %80 for all datasets.

- Study [Shi, Wan, Gao, Wang, (2018)] [7] combines Genetic Algorithm with Tabu Search, adresssed as GATS. Study states that Tabu Search meets the weaknesses of Genetic Algorithm and vice versa. Stated weaknesses are remature convergence for GA and initial solution dependence for TS. While GA provides quality initial solution to TS, TS helps GA to escape from local optimums. In this approach, algorithm starts with selecting features and cross-over. If early maturity occurs, individuals are sorted by their fitness values, top half is searched by Tabu Search, rest is mutated.

  Two datasets is used, containing +200 features. Comparison is made between proposed algorithm, standard GA, a multistart TS approach and classical ReliefF. For CPU time, GATS turned out to be the highest among others, but number of features after selection is lower and fitness is significantly higher. Study concludes that GATS method could improve the classification accuracy by presenting feature subsets with the within-class distances as small as possible and the between-class distances as big as possible.

- Paper [Beheshti, Demirel, Matsuda, (2017)] [9] introduces a computer-aided diagnosis (CAD) system that uses feature-ranking and a genetic algorithm. The Fisher criterion used as part of the objective function in the genetic algorithm. The Fisher criterion helps to select the optimum subset of features with maximum separation between two groups, to find the most discriminative features as an optimal subset. Results are argued with 15 other studies.

- In [Ahn, Hur, (2020)] [11], study is on time series classification in time-sensitive applications. A mathematical model for which the goal is to maximize the performance of the classification and minimize the classification start time and execution time is introduced.

5 large datasets are used. Comparison is held between well-known time series classifiers to confirm the classification performance, where one classifier is trained with the features chosen by the proposed GA, and another is trained with the features of general. 9 out of 70 cases showed that the general GA had better performance than the proposed GA.

- In [Goswami, Chakrabarti, Chakraborty, (2018)] [12], paper defined the he problem of finding optimal feature subset as a multi objective problem. The concept of redundancy is further refined with a concept of threshold value. Additionally, an objective of maximizing entropy has been added.

  Study uses 33 publicly available dataset. Paper compares 5 different methods varying over single objective and multi-objective approaches. A 12% improvement in classification accuracy is reported. The performance improvement is statistically significant as found by pair wise t-test and Friedman's test.

- In [Malakar, Gnosh, Bhowmik, Sarkar, Nasipuri, (2020)] [13], a Genetic Algorithm-based hierarchical feature selection (HFS), a wrapper FS method, model has been designed to optimize the local and global features extracted from each of the handwritten word images under consideration.

  Proposed model reduces the feature dimension by nearly 28%, but also enhances the performance of the handwritten word recognition (HWR) technique by 1.28% over the recognition performance obtained with unreduced feature set.

There are some nature-inspired optimizers such as Grey Wolf Optimizer, Ant Colony Optimizer, Artificial Bee Colony Optimizer etc. These optimizers are inspired by nature as said, and follow a natural approach that is in wild life. Genetic Algorithm can also be combined with these optimizers.

- In [Al-Tashi, Kadir, Rais, Mirjalili, Alhussian, (2019)] [6], a binary version of the hybrid grey wolf optimization (GWO) and particle swarm optimization (PSO) is proposed to solve feature selection problems. In GWO, four levels are considered in this algorithm. Three best solutions that have been found until a certain iteration is kept and forces others to modify their positions in the decision space consistent with the best place.

In PSO, particles are represented with a position vector and a velocity vector. Every particle has individual intelligence and search a search space around the best solution that it has found so far. Tests are driven with 18 datasets of different sizes and dimensions. To find the best solution, the K-nearest neighbors classifier with Euclidean separation matrix is utilized. The average of classification accuracy, fitness function, reduced feature, and computational time for all 18 datasets using the proposed method are achieved as: %93 accuracy, 0.073, 15.88 attributes, 7.76 seconds.

- In [Abukaligah, Khader, (2017)] [8], study proposes a hybrid of particle swarm optimization algorithm with genetic operators for the feature selection problem. Method finds accurate clusters by generating a new subset of more informative features. Method converts the text documents into a numerical matrix and implement PSO and GA to eliminate uninformative text features. K-means algorithm is used to obtain optimal clusters. The mean absolute difference (MAD) is used as a fitness function. It is based on the weighting scheme for evaluating the solution features.

  According to the evaluation measurements, the proposed feature selection method improved the efficiency of text clustering in almost all datasets. A thorough validation of the time analysis of the computational execution shows that the suggested method reported the shortest time and obtained accurate clusters.

- Another study, [Aalei, Shahraki, Rowhanimanesh, Eslami, (2016)] [10], particle swarm optimization (PSO) algoorithm is used in evaluation function, besides GA. Goal of the proposed model is selecting the best subset of features that can produce the highest classification accuracy for diagnosis and prognosis the breast cancer. Three datasets are used. The results show that feature selection could boost accuracy of classifiers. It is also clear from the results obtained that the selection of features has relatively increased specificity and sensitivity.

- In [Arabasadi, Alizadehsani, Roshanzamir, Moosaei, Yarifard, (2017)] [14] proposed a hybrid method that combines genetic algorithm and neural network. For feature selection, research considered four ranking methods, namely GINI-Index, weight by SVM, information gain and principal component analysis (PCA). The proposed method is able to increase the performance of neural network by approximately 10% through enhancing its initial weights using genetic algorithm which suggests better weights for neural network.

3 Genetic Algorithm

From the beginning of existence, all living creatures evolved and mutated partially or completely in many ways, and selected naturally over time. Meanwhile in this process, some features evolved to contribute to positive evolution, while some other features resulted in negative evolution. Individuals that had the positive evolution features increased their chance of survival, chance of breeding and chance of perpetuating the lineage while negatively mutated individuals fail to complete their life cycle and breed. In the absence of reproduction, these failed mutations died out over time.

Genetic Algorithm is a population-based metaheuristic method that is inspired by the natural selection process. It is a strong algorithm that is commonly used for optimization and search problems. There are five important steps of Genetic Algorithm that follows as: initial population, fitness function, selection, cross-over, mutation. These steps will be discussed below.

3.1 Initial Population

In GA, initial population contains individuals, which are referred as chromosomes. Attribute values are referred as genes. Set of chromosomes creates the initial population. In this paper, initial population is generated from the dataset by randomly selecting %50 of the instances.

3.1.1 Datasets

5 of different artificial databases are used. These datasets vary on difficulty on classification. Features that give the most information are known prior to the experiments. Those features are given in dataset descriptions and will be discussed in [Section 4.2]. This information is derived using Information Gain, GINI Decrease, ANOVA scoring methods.

To be able to see the efficiency of the proposed FS method on real datasets, 2 more large datasets taken from UCI ML Repository is used.

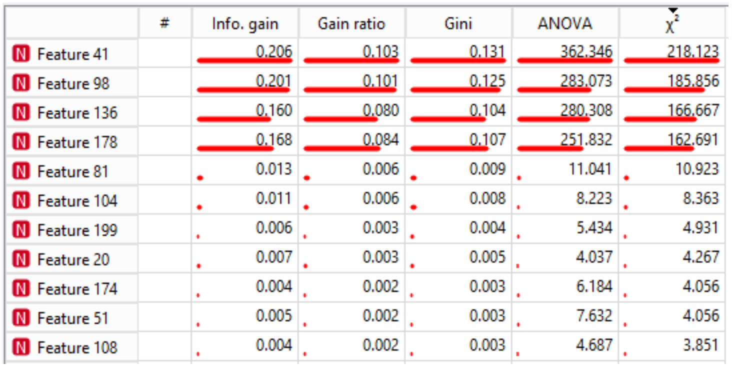
Data Set	Instances	Attributes	Num. of classes
TVD	120	785	6
PDD	756	755	2
AD1	1000	201	2
AD2	1000	201	5
AD3	1000	1001	3
AD4	1000	201	5
AD5	100	201	3

Table 1: Selected data sets.

Telugu Vowel Dataset There are just 26 letters in English literature, yet there are 1,924 in Telugu (Achulu (Vowels) – 16, Hallulu (consonants) – 36, Othulu – 36, and Guninthamulu – 34*16=544). As a result, Telugu character identification is more difficult than English. This dataset contains handwritten Telugu characters data, with six vowels of the Telugu language. 6 labels represent 6 vowels, linking every instance to one of the vowels.

Parkinson's Disease Dataset The information was acquired from 188 Parkinson's disease patients ranging in age from 33 to 87 at Istanbul University's Cerrahpasa Faculty of Medicine's Department of Neurology. To retrieve clinically valuable insights for PD diagnosis, several speech signal processing techniques were used for the speech recordings of Parkinson's Disease (PD) patients. Labels indicate diagnosis, where "1" denotes PD exists and "0" denotes no existence of disease.

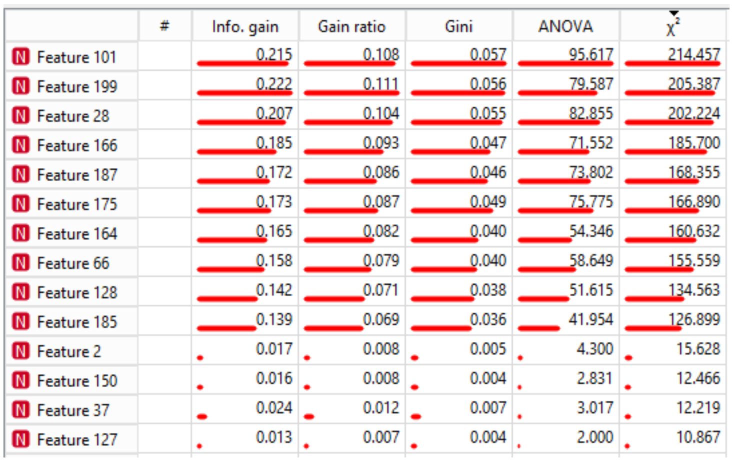
Artificial Database 1 This dataset is one of the artificial databases that are created. Dataset contains 200 attributes and 1 class attribute, with 1000 instances. The features that give out the most information are: Feature 41, Feature 98, Feature 136, Feature 178.



	#	Info. gain	Gain ratio	Gini	ANOVA	χ^2
Feature 41		0.206	0.103	0.131	362.346	218.123
Feature 98		0.201	0.101	0.125	283.073	185.856
Feature 136		0.160	0.080	0.104	280.308	166.667
Feature 178		0.168	0.084	0.107	251.832	162.691
Feature 81		0.013	0.006	0.009	11.041	10.923
Feature 104		0.011	0.006	0.008	8.223	8.363
Feature 199		0.006	0.003	0.004	5.434	4.931
Feature 20		0.007	0.003	0.005	4.037	4.267
Feature 174		0.004	0.002	0.003	6.184	4.056
Feature 51		0.005	0.002	0.003	7.632	4.056
Feature 108		0.004	0.002	0.003	4.687	3.851

Figure 2: AD1 Feature Information

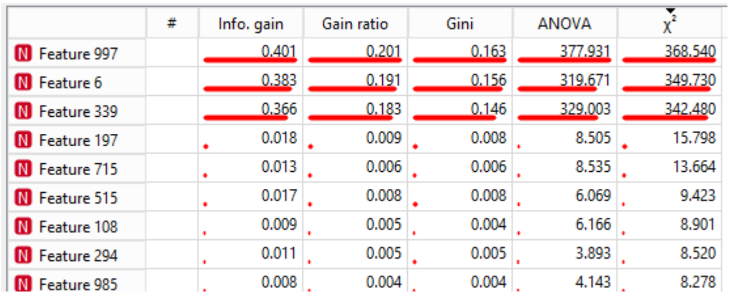
Artificial Database 2 This dataset is one of the artificial databases that are created. Dataset contains 200 attributes and 1 class attribute, with 1000 instances. The features that give out the most information are: Feature 28, Feature 66, Feature 101, Feature 128, Feature 164, Feature 166, Feature 175, Feature 185, Feature 187, Feature 199.



	#	Info. gain	Gain ratio	Gini	ANOVA	χ^2
Feature 101		0.215	0.108	0.057	95.617	214.457
Feature 199		0.222	0.111	0.056	79.587	205.387
Feature 28		0.207	0.104	0.055	82.855	202.224
Feature 166		0.185	0.093	0.047	71.552	185.700
Feature 187		0.172	0.086	0.046	73.802	168.355
Feature 175		0.173	0.087	0.049	75.775	166.890
Feature 164		0.165	0.082	0.040	54.346	160.632
Feature 66		0.158	0.079	0.040	58.649	155.559
Feature 128		0.142	0.071	0.038	51.615	134.563
Feature 185		0.139	0.069	0.036	41.954	126.899
Feature 2		0.017	0.008	0.005	4.300	15.628
Feature 150		0.016	0.008	0.004	2.831	12.466
Feature 37		0.024	0.012	0.007	3.017	12.219
Feature 127		0.013	0.007	0.004	2.000	10.867

Figure 3: AD2 Feature Information

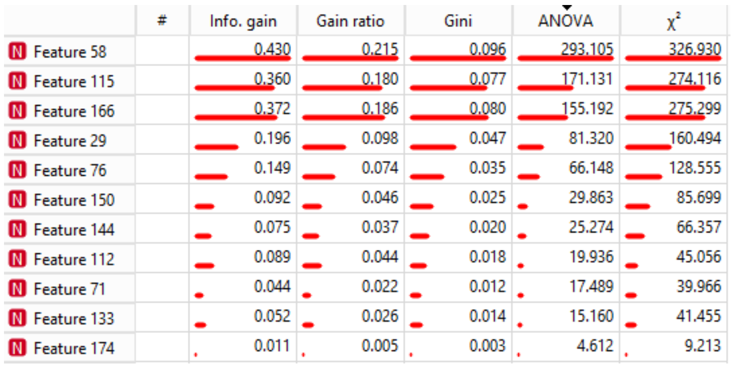
Artificial Database 3 This dataset is one of the artificial databases that are created. Dataset contains 1000 attributes and 1 class attribute, with 1000 instances. The features that give out the most information are: Feature 6, Feature 339, Feature 997.



	#	Info. gain	Gain ratio	Gini	ANOVA	χ^2
Feature 997		0.401	0.201	0.163	377.931	368.540
Feature 6		0.383	0.191	0.156	319.671	349.730
Feature 339		0.366	0.183	0.146	329.003	342.480
Feature 197		0.018	0.009	0.008	8.505	15.798
Feature 715		0.013	0.006	0.006	8.535	13.664
Feature 515		0.017	0.008	0.008	6.069	9.423
Feature 108		0.009	0.005	0.004	6.166	8.901
Feature 294		0.011	0.005	0.005	3.893	8.520
Feature 985		0.008	0.004	0.004	4.143	8.278

Figure 4: AD3 Feature Information

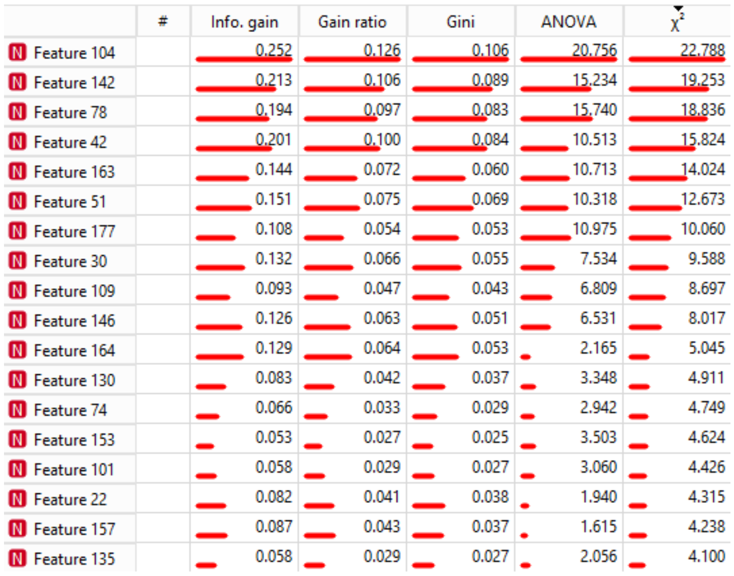
Artificial Database 4 This dataset is one of the artificial databases that are created. Dataset contains 200 attributes and 1 class attribute, with 1000 instances. The features that give out the most information are: Feature 29, Feature 58, Feature 71, Feature 76, Feature 112, Feature 115, Feature 133, Feature 144, Feature 150, Feature 166.



	#	Info. gain	Gain ratio	Gini	ANOVA	χ^2
Feature 58		0.430	0.215	0.096	293.105	326.930
Feature 115		0.360	0.180	0.077	171.131	274.116
Feature 166		0.372	0.186	0.080	155.192	275.299
Feature 29		0.196	0.098	0.047	81.320	160.494
Feature 76		0.149	0.074	0.035	66.148	128.555
Feature 150		0.092	0.046	0.025	29.863	85.699
Feature 144		0.075	0.037	0.020	25.274	66.357
Feature 112		0.089	0.044	0.018	19.936	45.056
Feature 71		0.044	0.022	0.012	17.489	39.966
Feature 133		0.052	0.026	0.014	15.160	41.455
Feature 174		0.011	0.005	0.003	4.612	9.213

Figure 5: AD4 Feature Information

Artificial Database 5 This dataset contains 200 attributes and 1 class attribute, with 100 instances. There are many features that give out information. Therefore, this dataset is hard to distinguish most valuable features. 5 of the most valuable features are: Feature 42, Feature 78, Feature 104, Feature 142, Feature 163; but the list goes on long.



	#	Info. gain	Gain ratio	Gini	ANOVA	χ^2
Feature 104		0.252	0.126	0.106	20.756	22.788
Feature 142		0.213	0.106	0.089	15.234	19.253
Feature 78		0.194	0.097	0.083	15.740	18.836
Feature 42		0.201	0.100	0.084	10.513	15.824
Feature 163		0.144	0.072	0.060	10.713	14.024
Feature 51		0.151	0.075	0.069	10.318	12.673
Feature 177		0.108	0.054	0.053	10.975	10.060
Feature 30		0.132	0.066	0.055	7.534	9.588
Feature 109		0.093	0.047	0.043	6.809	8.697
Feature 146		0.126	0.063	0.051	6.531	8.017
Feature 164		0.129	0.064	0.053	2.165	5.045
Feature 130		0.083	0.042	0.037	3.348	4.911
Feature 74		0.066	0.033	0.029	2.942	4.749
Feature 153		0.053	0.027	0.025	3.503	4.624
Feature 101		0.058	0.029	0.027	3.060	4.426
Feature 22		0.082	0.041	0.038	1.940	4.315
Feature 157		0.087	0.043	0.037	1.615	4.238
Feature 135		0.058	0.029	0.027	2.056	4.100

Figure 6: AD5 Feature Information

3.2 Fitness Function & Accuracy

Fitness function is one of the most important properties of the GA. This function determines how fit the given individual is, so that the algorithm can decide whether this individual should breed or not later on selection process.

In this study, aim is to select minimum number of features that serves the most relevance to target. Firstly accuracy is calculated with the help of logistic regression. To reach the desired reduced number of features, selection pressure is needed. This pressure is provided by calculating fitness with accuracy over square root of number of features selected, as given below.

$$Fitness = accuracy/\sqrt{numberoffeaturesselected}$$

Even though the fitness is calculated through accuracy, the most fit and most accurate solutions may differ because of the effect of the number of selected features. Both the most accurate and most fit solutions are valuable to serve the aim. Therefore, accuracies and fitnesses are tracked individually to be able to observe the variance through iterations and to be able to protect the most fit and most accurate solutions from mutation and cross-over effects.

3.3 Selection

To create the new population, individuals from the existing population to be crossed and mutated must be selected. According to the theory, individuals who are good should continue their lives and new individuals should be created from these individuals. Therefore, individuals with higher fitness values are more likely to be selected. Usually, the worst chromosomes are replaced when new chromosomes are added to the population. The most known selection methods are Roulette Wheel Selection, Tournament Selection and Random Selection.

In selection, some chromosomes are selected with tournament selection to create the parent pool of the next generation. Chromosomes with higher fitness values has the higher chance of reproduction.

Tournament Selection To put a selection pressure on the system and to individuals, Tournament Selection is used to select the parents of the next generation. Chosen at random from the chromosome pool, Tournament Selection method runs multiple "tournaments" on those chromosomes to select the most fit among them. The champion of each tournament is chosen to cross-over.

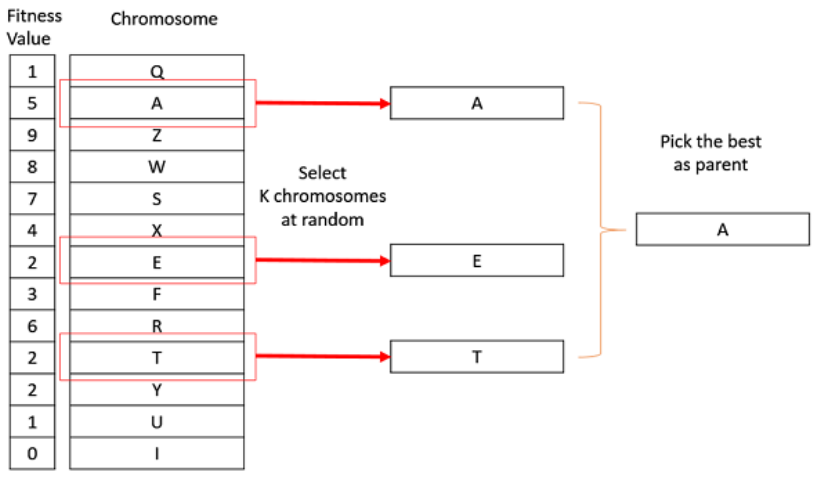


Figure 7: Tournament selection

3.4 Main GA Functions

Since GA is an evolutionary algorithm, the production of new individuals from parents is crucial to get closer to the optimal/best answer through iterations. Main GA functions are important to keep the population varied in generations. These functions include different kind of cross-over methods and mutations which will be discussed later.

Before applying the main GA function to population; population, fitness and accuracy values are sorted with respect to fitness values to be able to perform special methods on least fit individuals and to protect most fit solutions.

3.4.1 Cross-over

Cross-over is the most discriminating phase in GA, than other FS algorithms. Cross-over guarantees that genetic information is exchanged across parents, resulting offspring chromosomes to be more likely superior to the parents.

To explain this with very basic example we can think to choose a random break point (the cross-over point) and combine everything before that point from the first ancestor and everything after from the second ancestor to create the offspring. How many chromosomes will be crossed from a problem solution space is determined according to the cross-over ratio P(c). In this paper, asexual reproduction (cloning an existing solution) is not used.

The cross-over method is used to determine or examine the available potentials of genes. However, if the population does not contain all the encoded information, the cross-over method cannot produce the expected solution. Solution for this argument is presented in [Section 3.4.2] .

Two-point Cross-over To increase diversity, two-point cross-over is implemented. With the help of selection process and tournament selection, two parents are selected from. the parent pool. Two points which represent the two points of chromosome to perform cross-over is chosen randomly. Then, first individual and second individual trade the genes between two points and create an offspring. New offspring is added to new population of next generation.

Cross-over is applied to %30 of the population.

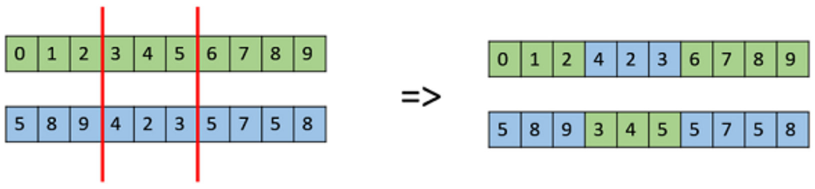


Figure 8: Cross-over

Min-Max Cross-over To increase the fitness faster through generations, a custom cross-over method is developed. This method works similar to a basic two-point cross-over, generating an offspring using the least and most fit individuals, and replacing it with the least fit parent.

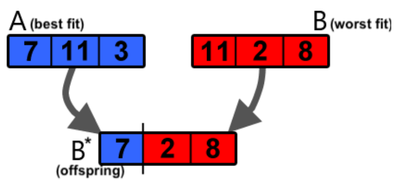


Figure 9: Min-max cross-over

3.4.2 Mutation

Mutation is simply an operator that makes changes on chromosomes' own genes or small units that make up genes. The main purpose of mutation is to maintain genetic diversity in the population. With mutation, some probabilities that are not present or low can have a chance to join and effect the evolutionary process of the individuals. The advantage of mutation in GA is to prevent the research from entering a vicious circle by providing direction changes in the search for the solution area of the problem.

How many chromosomes to mutate in a problem pool is decided according to the mutation rate P(m). In general, the mutation probability is kept low. Therefore, mutation effects are less common in chromosomes. During mutation, the number of genes in the chromosome does not change, rather remains constant. If the probability of mutation increases, the genetic search becomes an undesired random search. But it helps in finding lost genetic material again, which explains why it is necessary.

Scrambled Mutation With a low probability, given as 0.2 in the paper, mutation is applied to the 20 percent of the population. From the population, a random chromosome is selected. Then, from the entire chromosome, a subset of genes is selected randomly and the values they hold scrambled randomly.

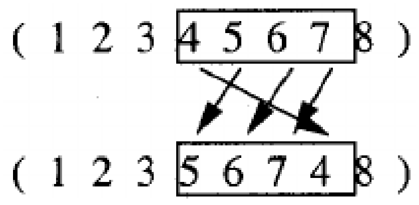


Figure 10: Scrambled Mutation

3.5 Feature Relevance/Importance

The term "feature relevance" refers to a set of methodologies for providing scores to input attributes in a predictive model, representing the relative significance of each feature while making a prediction. This attribute of Logistic Regression can show the relevance of the features to each other, which can help determine feature pairs or multiple features that carry information together, and also relevance to label class.

These scores are beneficial in a variety of circumstances in predictive modeling problems, such as better comprehending the data, better comprehending a model, and lowering the number of input attributes.

In this paper, relevances are derived using Logistic Regression technique. Feature relevance is used to make the algorithm even more selective and to put more pressure on selection process by eliminating the features that have lowest relevance to target class.

4 Experiment

4.1 Environment

GA is coded on Python environment, 4.0.1.

Pandas is a Python library that is used in data science and data analysis. this library provides DataFrame object, which is used for initial population in this experiment.

NumPy, standing for Numerical Python, is another Python library that is used for computing on array structures, can also be used for Fourier calculations, matrices etc. In this experiment, NumPy is used to work with parent and offspring arrays.

Python Math module is one of the library's that is used in this experiment for mathematical tasks.

Python Scikit-Learn (sklearn) library is one the most powerful libraries in Python. This library is used for ML and modelling such as classification, regression, clustering etc. In this experiment, this library is used for Logistic Regression, train-test split methods and to calculate the accuracy scores of populations.

Mat Plot Library's pyplot (matplotlib.pyplot) in Python contains functions that make matplotlib work like MATLAB. pyplot functions are used to create figures and graphs, plotting lines and updating figures and lines.

Seaborn is another Python library that is used for data visualization, based on matplotlib. This library is used for drawing informative graph and keeping output data visualized, through generations and trials.

4.2 Tests

Algorithm is constructed and ran with 0.2 mutation rate. 10 independent tests are ran on datasets with n=10 solutions and 100 iterations. Different tests result in varying accuracies and number of selected features. Results are discussed individually, based on average results and best results on 10 independent tests.

Dataset	**Accuracy**
TVD	% 83.86
PDD	% 80.74
AD1	% 82.66
AD2	% 79.63
AD3	% 97.66
AD4	% 92.99
AD5	% 92.33

Table 2: Average results of tests.

Dataset	**Selected num. of features**	**Accuracy**
TVD	14	% 83.8
PDD	756	755
AD1	1000	201
AD2	1000	201
AD3	1000	1001
AD4	1000	201
AD5	100	201

Table 3: Best results of tests.

Average and best results depending on the databases that tests ran on are discussed below.

TVD Experiment While tests on The Vowel Dataset resulted in average of %83.86 accuracy, Trial 7 has the highest accuracy score among all. After 100 iterations, best trial selected 4 features with % 99.4 accuracy. Execution took 2979 seconds.

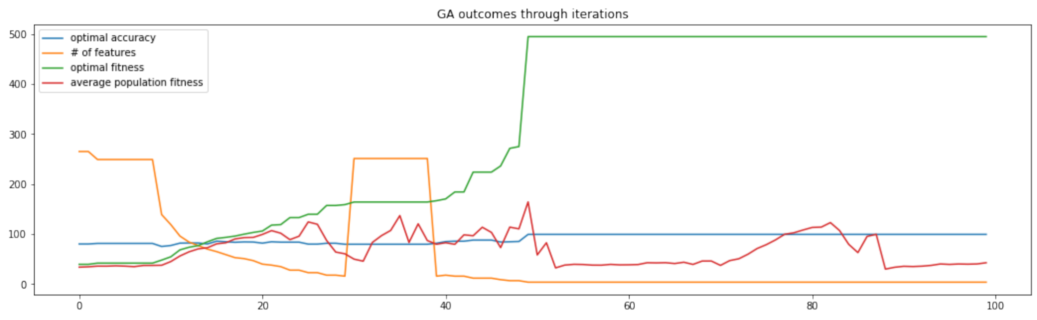


Figure 11: Telegu Vowel Dataset performance graph through 100 iterations

PDD Experiment Before running the algorithm on Parkinson's Disease Dataset, some missing values (represented as NaN) was filled with average values of respected columns.

Among 10 tests, Trial 2 gave the best solution with execution time as 100 seconds. Selected number of features waas 12, namely: Feature 154, 157, 158, 288, 289, 290, 394, 397, 400, 404, 405, 406. Respected accuracy is given as %84.5.

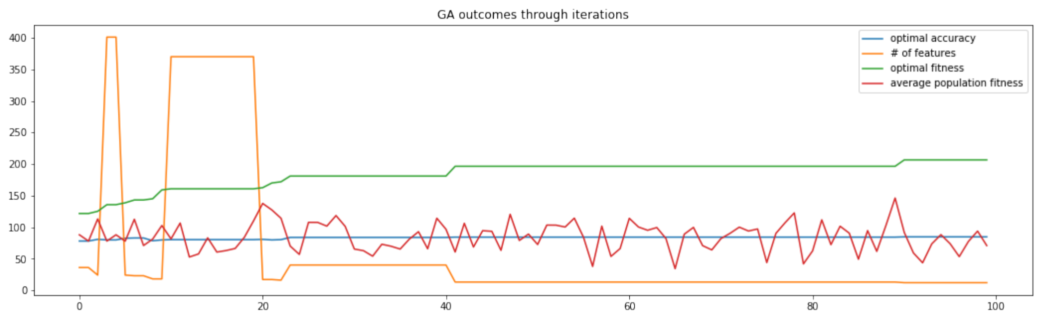


Figure 12: Parkinson's Disease Dataset performance graph through 100 iterations

AD1 Experiment Below graph shows results in different tests. As seen on graph, selected number of features vary between 0-10 region.

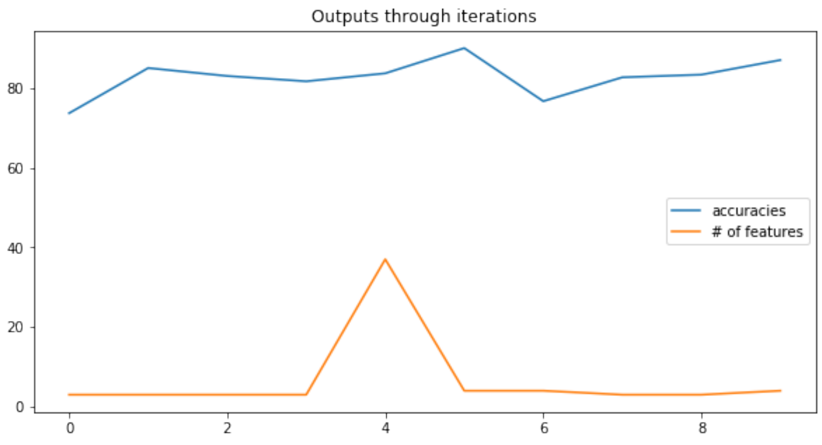


Figure 13: AD1 outputs graph through trials

AD1 have 4 features that carries the most information, namely: Feature 41, 98, 136 and 178. Among independent trials, Trial 5 has the best results. With execution time as 24 seconds, algorithm selected 4 features (Feature 41, Feature 96, Feature 178, Feature 199) with %90.0 accuracy. Below graph shows the performance of the algorithm through iterations.

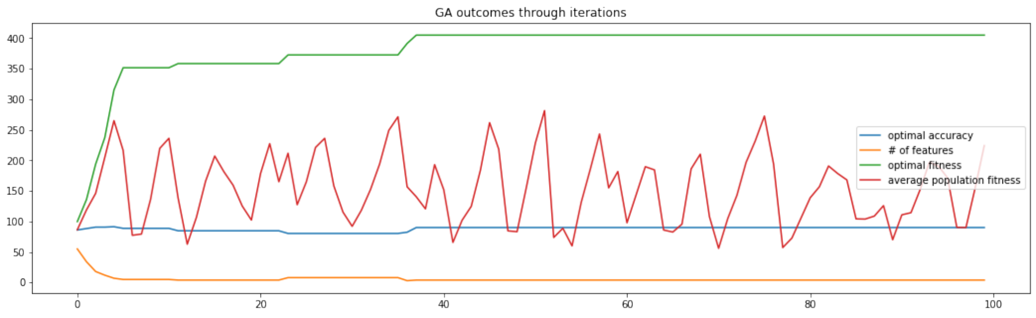


Figure 14: AD1 performance graph through 100 iterations

AD2 Experiment Below graph shows how varying the results through runs can be, especially for number of features selected. Number of selected features peaked in Trial 6 and Trial 8, while accuracy showed a more consistent structure.

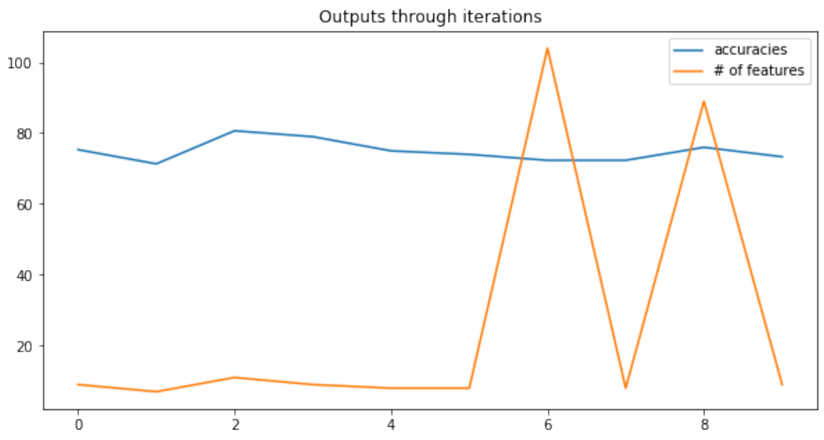


Figure 15: AD2 outputs graph through trials

AD2 have 10 features that carries the most information, namely: Feature 28, 66, 101, 128, 164, 166, 175, 185, 187, 199. Among independent trials, Trial 2 has the best results derived in 25 seconds. With execution time as 67 seconds, algorithm selected 11 features (Feature 28, 66, 97, 101, 128, 164, 166, 175, 185, 197, 199) with %80.6 accuracy.

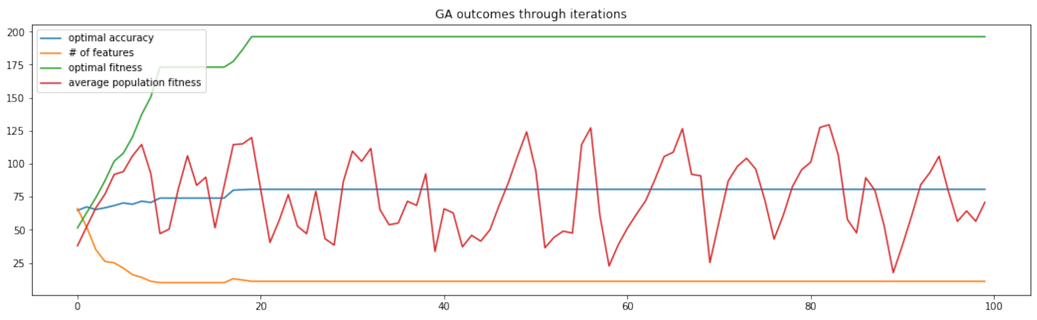


Figure 16: AD2 performance graph through 100 iterations

AD3 Experiment AD3 experiment results shows a consistent structure on accuracy through trials. Number of selected features peaked in Trial 7. It can be seen that even the number of selected features peaked or resulted as minimum, accuracy stayed close to %100, showing how small of a percentage additional features add to the overall accuracy.

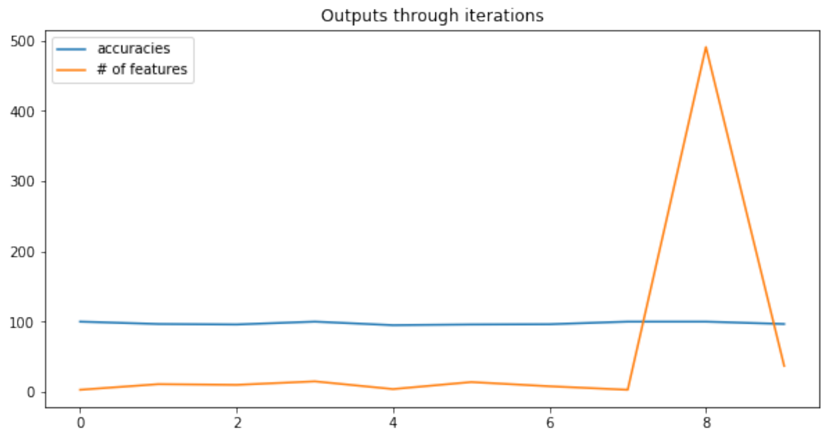


Figure 17: AD3 outputs graph through trials

AD3 have 3 base features that carries out the most information, namely: Feature 6, 339, 997. The best result is obtained at the end of Trial 6, with execution time 58 seconds. Algorithm selected 8 features which are Feature 6, 95, 97, 276, 339, 357, 1000. Final accuracy is % 96.3 with execution time as 172 seconds.

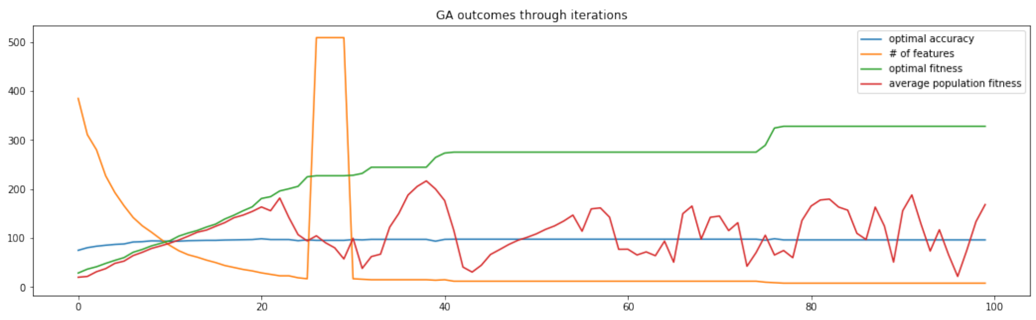


Figure 18: AD3 performance graph through 100 iterations

AD4 Experiment AD4 experiment results show a peak at Trial 6 while accuracy gets lower, around %80 region. This may be caused by noisy data that adding more features may create more noise and lower the accuracy.

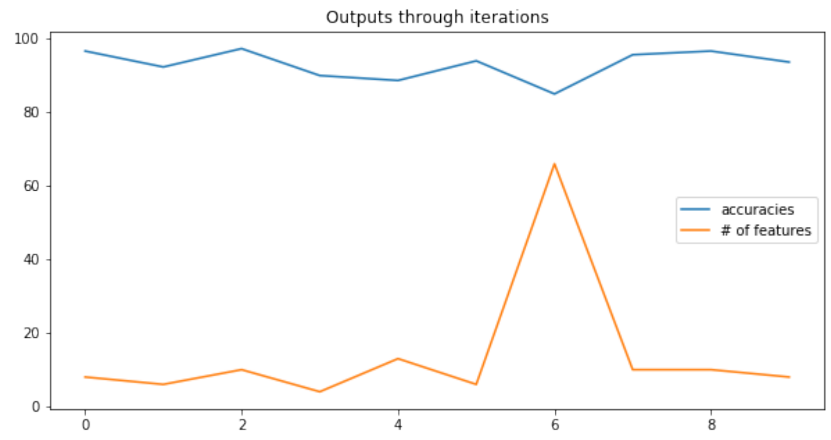


Figure 19: AD4 outputs graph through trials

AD4 carries most of it's information on 10 features, namely: Feature 29, 58, 71, 76, 112, 115, 133, 144, 150, 166. Among independent tests, Trial 2 has the best results in terms of accuracy. Algorithm ran for 81.3 seconds, and selected 9 features with %97.3 accuracy. Selected features are listed as: Feature: 42, 58, 71, 76, 115, 150, 166, 189, 200.

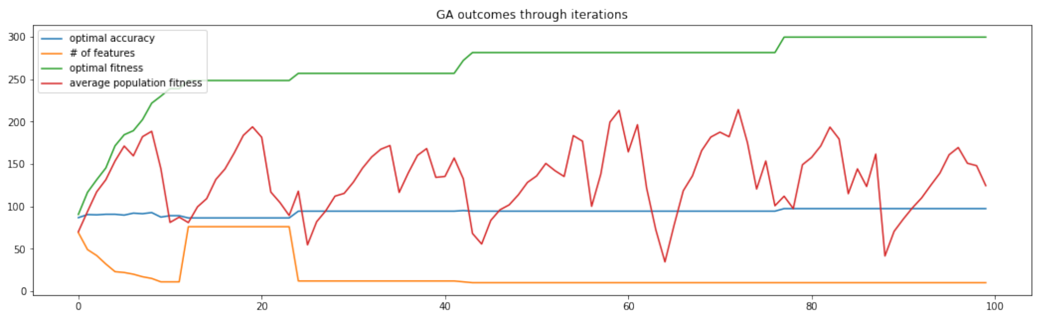


Figure 20: AD4 performance graph through 100 iterations

AD5 Experiment Graph of AD5 trials shows the most consistent structure among all. None of the trials showed a peak up or down on neither number of features selected, nor accuracy scores.

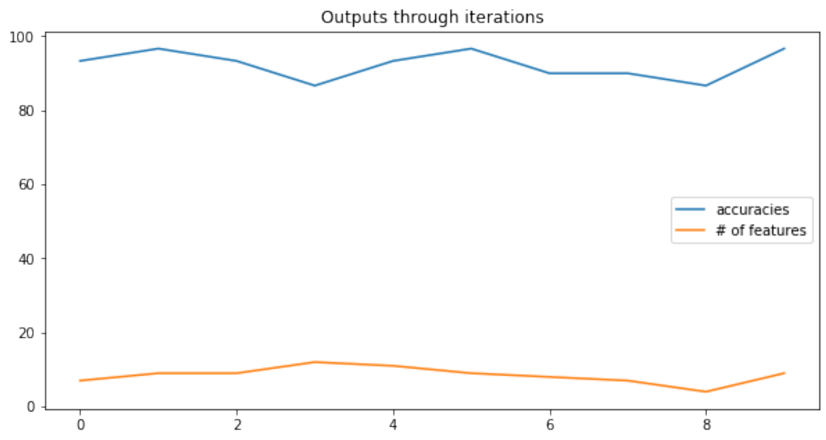


Figure 21: AD5 outputs graph through trials

AD5 carries it's valuable information scattered over large number of features, according to other datasets that's been used. As a result, 3 of the tests ran on AD5 showed similar outcomes, selecting 9 features with accuracy %96.6. Selected features in these trials are given below.

Selected features on Trial 1: Feature 15, 125, 130, 142, 176, 181, 193, 199, 200.

Selected features on Trial 5: Feature 6, 9, 62, 142, 163, 166, 174, 195, 200.

Selected features on Trial 9: Feature 42, 104, 109, 130, 163, 170, 177, 196, 200. Below is the graph of Trial 9 through iterations. Execution time is noted as 14 seconds.

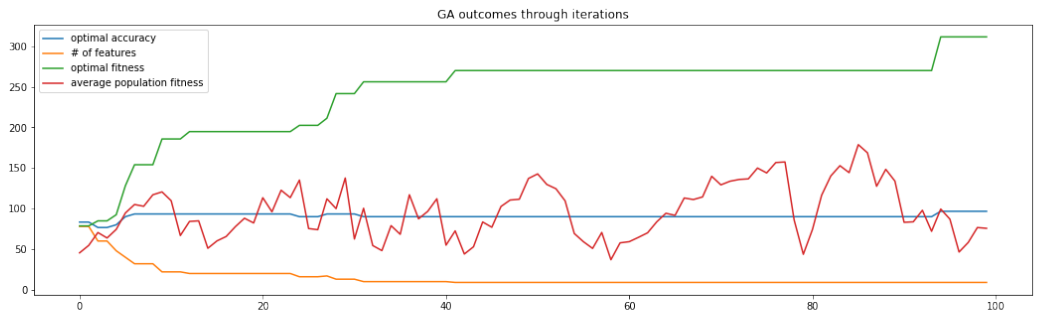


Figure 22: AD5 performance graph through 100 iterations

5 Discussion & Conclusion

Through tests, algorithm has shown consistent results in terms of accuracy. Reaching accuracies higher than %75 nearly in ever test shows how efficient this algorithm works. With test results that reach even higher accuracies with very low number of features, it is concluded that with the help of proposed algorithm, computational cost and time complexity of further operations can be notably reduced.

Even though results vary, with the best solution graphs discussed earlier, potential of the Genetic Algorithm over Feature Selection can be deduced.

Since this algorithm works on datasets with large number of attributes, one of the constraints this algorithm might face is RAM and device capacity. If the dataset is too large to store or execute operations on, algorithm can not be applied. To eliminate this constraint, a larger RAM or memory space might be needed. Another constraint this algorithm may face is time complexity. Even though algorithm aims to reduce the time complexity for the subsequent operations, it still needs to work on the whole dataset through iterations. If the dataset is small enough to fit the memory, but large enough to be unsolvable in a feasible time algorithm can not be applied. To be able to apply the proposed algorithm, one of the less effective -but relatively fast-feature selection operations might be needed prior to the GA to reach the feasible region.

Timely cost of this algorithm may vary and depend on size and complexity of dataset. Observed time variation during tests is noted as between 10 and 3000 seconds. Since the algorithm allocates a memory and uses resources during this time period, device that runs the algorithm consume some amount of electricity to stay on. A PC consumes 200 Watt hours (Wh) on average, meaning the electricity cost of this algorithm as observed is between 0.5 Watts and 166.6 Watts. This cost can be calculated as currency with respect to electricity price per watt of the region.

This paper addresses the IEEE Standards Association's (IEEE SA) IEEE P7003™—Algorithmic Bias Considerations standards. With that, it is declared here that in order to minimize unjustifiable discriminatory effects on users, most up-to-date and latest practices are employed in the design, testing, and analysis of this algorithm.

To meet the Coding Standards and Guidelines, global variables are used limitedly. Also, in the code, indentations are used widely to keep code understandable and simple by increasing the readability of the code.

6 Obstacles

While developing, it was noted that fitness function gave varying results through iteration. While this is expected and normal, having fitness decreasing between iterations is undesired because that means with the evolutionary nature of Genetic Algorithm, best fitness achieved is not protected and can be lost through iterations. To prevent this, elitism approach is adopted. Elitism refers to the concept that the best fit individual or group of individuals is assured a spot in the upcoming generation - usually without the priority for mutation. In addition to being carried forward, they should still be eligible to be chosen as parents. By adopting this approach, solution with best fitness value yet to be achieved is stored safely through iterations and protected unless a better fit individual is breed.

Another obstacle while developing and testing this algorithm was to determine the optimum value of number of individuals to run algorithm with, denoted as "n" earlier on the paper. It is concluded by testing, having a higher number of individuals may decrease the time between breeding the fitter individuals. Because with more individuals, cross-overs vary wider among individuals. On the contrary, having higher number of individuals to run the algorithm with increases overall execution time, since more cross-overs and possible mutations are executed. With this in mind, "n" determined and fixed as 10 for testing purposes.

Similar to "n", number of iterations was another variable that needed to be determined to run the tests. While higher number of iterations may create a better chance to find a fitter individual, with mutation probability and evolutionary nature of Genetic Algorithm, this may never happen. Since there is no optimum number of iterations that fits to every problem, this number is determined and fixed as 100 for testing purposes.

Genetic Algorithm is a nature-based evolutionary algorithm. Therefore, the chance of reaching higher fitness values through iterations is aleatory. Fixing the number of individuals to work with and number of iterations may result in low accuracy scores or high number of selected features in runs. To avoid this and demonstrate the true potential of Genetic Algorithm over Feature Selection, proposed algorithm is executed multiple times to be able to see the better results that can be obtained.

7 Future Work

After building the proposed GA and experimenting on varying datasets, results were sufficient. Even though best results were satisfying, it is observed that in some tests algorithm selects large number of features, failing to eliminate significant number of features. Since GA is an evolutionary algorithm, these results are understandable; yet undesirable. In the feature, some new or existing methods can be developed to put more selection pressure over population to derive more consistent results in terms of number of features selected.

In future work, algorithm can be observed more widely with more testing and under varying constants such as population size and number of iterations. A stronger device can be used to run the code with larger datasets to observe the results on dataset sizes that are not covered in this paper.

Feature selection with Genetic Algorithm can also be extended by hybridizing the Genetic Algorithm with one of the nature-inspired optimizers mentioned earlier, or other feature selection methods or optimizers.

References

[1] Raymer, M. L., Punch, W. F., Goodman, E. D., Kuhn, L. A., Jain, A. K. (2000). Dimensionality Reduction Using Genetic Algorithms. IEEE Transactions on Evolutionary Computation, 4 (2), 164-171.

[2] Babatunde, O. H., Armstrong, L. , Leng, J. , Diepeveen, D. (2014). A Genetic Algorithm-Based Feature Selection. International Journal of Electronics Communication and Computer Engineering, 5(4), 899-905.

[3] Yang, Jihoon and Honavar, Vasant, "Feature Subset Selection Using a Genetic Algorithm" (1997). Computer Science Technical Reports. 156.

[4] Swati Jadhav, Hongmei He and Karl Jenkins (2018). Information Gain Directed Genetic Algorithm Wrapper Feature selection for Credit Rating. Appliied Soft Computing, Volume 69, 541-553

[5] H. Lu et al., A hybrid feature selection algorithm for gene expression data classification, Neurocomputing (2017)

[6] Al-Tashi, Kadir, Rais, Mirjalili, Alhussian, (2019). Binary Optimization Using Hybrid Grey Wolf Optimization for Feature Selection.

[7] Lei Shi, Youchuan Wan, Xianjun Gao, Mingwei Wang, "Feature Selection for Object-Based Classification of High-Resolution Remote Sensing Images Based on the Combination of a Genetic Algorithm and Tabu Search", Computational Intelligence and Neuroscience, vol. 2018, Article ID 6595792, 13 pages, 2018.

[8] Abiualigah Laith, Khader Ahamad Tajudin, (2017). Unsupervised text feature selection technique based on hybrid particle swarm optimization algorithm with genetic operators for the text clustering.

[9] Iman Beheshtia, Hasan Demirel, Hiroshi Matsudaa, for the Alzheimer's Disease Neuroimaging Initiative1 (2017). Classification of Alzheimer's disease and prediction of mild cognitive impairment-to-Alzheimer's conversion from structural magnetic resource imaging using feature ranking and a genetic algorithm.

[10] Aalaei, S., Shahraki, H., Rowhanimanesh, A., Eslami, S., (2016). Feature selection using genetic algorithm for breast cancer diagnosis: experiment on three different datasets.

[11] Ahn, G., Hur, S., Efficient Genetic Algorithm for Feature Selection for Early Time Series Classification, Computers and Industrial Engineering (2020).

[12] Goswami, S., Chakrabarti, A., Chakraborty, B., An empirical study of feature selection for classification using genetic algorithm (2018).

[13] Malakar, S., Ghosh, M., Bhowmik, S., Sarkar, R., Nasipuri, M., A GA based hierarchical feature selection approach for handwritten word recognition (2020).

[14] Arabasadia, Z., Alizadehsanib, Al., Roshanzamir, M., Moosaeid, H., Yarifard, A. A., Computer aided decision making for heart disease detection using hybrid neural network-Genetic algorithm (2017).