CHAPTER 4

INFORMATION GAIN NRGA ALGORITHM FOR FEATURE SELECTION OF MICROARRAY DATA

4.1 INTRODUCTION

The proposed work incorporates microarray data as the input for process execution. The resultant data obtained from that technique is arranged as a row and column matrix, in which the column genes are represented as the whole genome. The rows of the matrix are symbolized as the various samples like various tissues, experimental conditions, or time points. From the given wealthy quantity of GE data, it is necessary to extract hidden information from the given matrix. The process initializes the population from the GE data.

Microarray gene expression technology has opened the possibility of investigating the activity of thousands of genes simultaneously. Gene expression profiles show the relative abundance of mRNA corresponding to the genes. Thus, discriminant analysis of microarray data has great potential as a medical diagnostic tool since results represent the state of a cell at the molecular level. The goal of microarray data classification is to build an efficient model that identifies the differentially expressed genes and may be used to predict class membership for any unknown samples. Challenges posed in microarray classification are the availability of only a limited number of samples in comparison to the high-dimensionality of the samples, and
experimental variations in measured gene expression levels. The classification of microarray data samples involves feature selection and classifier design. Generally, only a small number of gene expression data show a strong correlation with a certain phenotype compared to the total number of genes investigated. This means that of the thousands of genes investigated, only a small number show significant correlation with a certain phenotype. Consequently, in order to analyze gene expression profiles correctly, feature (gene) selection is crucial for the classification process. The goal of feature selection is to identify the subset of differentially expressed genes that are potentially relevant for distinguishing the sample classes. A good selection method for genes relevant for sample classification–based on the number of genes investigated–is needed to increase the predictive accuracy and to avoid incomprehensibility. Several methods have been used to perform feature election.

Before the feature selection is performed the ranking level for the random population is assigned based on sorting technique. Hence, selecting genes from the microarray data poses a terrible challenge due to their high-dimensionality features in clustering technique. A clustering algorithm is proposed, a hybrid model of IG NRGA algorithm for feature selection in microarray data sets. IG was used to select important feature subsets (genes) from all features in the GE data and NRGA was employed for actual feature selection. The KNN method is used to evaluate the NRGA algorithm.

4.2 FEATURE SELECTION FOR GENE EXPRESSION

Owing to the particular characteristics of GE data to the proposed work and the scrupulous requirements from the biological field, gene-based clustering gives several new challenges is still an open problem in finding the solution to accurate classification.
4.2.1 Challenges of Gene Clustering

Feature selection is an important data processing way in data mining field. With in-depth data mining study, the research object becomes more and more complex; the dimensional of object feature becomes more and more high. There are a lot of redundant features and noise characteristics in the feature space of high dimensional data. On one hand these features may reduce the classification or clustering accuracy, on the other hand it will greatly increase the space and time complexity of the learning and training. Therefore, in the face of high dimensional data analysis, it usually need to use feature selection algorithm to find feature subspace with good separability, thereby realizing the dimension reduction in data mining, reduce time and space complexity. Feature selection is an important step in knowledge discovery (KDD) process; it's very useful to pretreatment, mining, processing in KDD.

For each problem with some sample, there is a maximum number of features where performance degrades instead of improves – which is called the curse of dimensionality. An accurate mapping of lower-dimensional space of features is needed so no information is lost by discarding important and basic features. Two issues one should pay attention to while doing this: (1) How dimensionality can affect classification accuracy and (2) How dimensionality affects a classifier complexity. A feature is good when it is relevant but not redundant to the other relevant features. There are two techniques to follow for this: feature extraction and feature selection. Feature extraction algorithms tend to create a new subset of features by combining existing features. Feature selection (FS) algorithms tend to limit the features to only those which would improve a task performance. The FS is an essential machine learning technique that is important and efficient in
building classification systems. When used to reduce features, it results in lower computation costs and better classification performance.

Feature selection algorithms are composed of three components: search algorithm, evaluation function, and performance function. The search algorithm could be: exponential – which is expensive to use as they have exponential complexity in number of features, sequential where it adds and subtracts features, so they have polynomial complexity; or randomized – where it require biases to yield small subsets, and they usually achieve high accuracies. An objective function is a function to evaluate the candidate features for feature selection. Based on evaluation criteria, FS techniques can be divided into filter methods and wrapper methods. Filters evaluate feature subsets by their information content, using distance measures, correlation measures...etc. Wrappers use a classifier for features subset evaluation by their predictive accuracy. Filter techniques discards feature upon their evaluation based on data general characteristics or using some kind of statistical analysis, without any learning mechanism involved. Wrapper techniques use a learning algorithm to find the features subset with the best performance. They are more expensive computation-based, and slower due to the repeating process, but they give more accurate results than filter techniques. This might be a drawback for high dimensional data but it could be defeated by using a fast learning algorithm. Hence there exists a need of embedded efficient feature selection technique.

4.2.2 Gene Expression

A set of GE data can be examined as a table with rows indicating genes, columns indicating various samples. Later every position in the table explains about the dimension for a particular selected gene in the particular sample. It calls the table as a GE matrix for performing feature selection. In
addition to the matrix, a description of a microarray experiment should also contain information about the genes whose expression has been measured and the experimental condition under which the samples were taken to do ranking and sorting. The information required to describe a microarray experiment can be divided conceptually into three logical parts: gene annotation, sample annotation and a GE matrix.

Ideally it likes to measure amounts of GE in natural units, such as mRNA copies per cell and to have an error estimate such as the standard deviation associate with each value. The threshold value is set to perform the feature selection. There are a number of experimental challenges, however that, make direct measurement of GE difficult. Raw data from microarray experiments are images from hybridized microarray scans that have to identify and quantify each feature spot in the image. A DNA sequence are spotted on a microarray several times, several distinct DNA sequences that map to the same gene. To yield a single value for these, the corresponding measurements have to be combined. Moreover, the same biological condition can be measured in several replicate hybridizations and the information from all the replicates has summarized to derive a single GE data matrix. Finally, to compare microarray data from several samples, the data must be approximately normalized.

There are at least three levels of data relevant to a microarray experiment:

- The scanned images (raw data)
- The quantitative output from the image analysis procedure and
- The derived measurement
There is an important measurement series of transformations leading from raw data to the GE matrix and the steps involved are far from being standardized.

As there are no widely used standard controls for microarray data from different sources, it uses different measurement units whose conversion factors are unknown. It may vary depending on expression levels. It indicates the necessity to record not only the final GE matrix. But also a detailed description of how the expression values were obtained, if verification of the data is to be ensured. Consequently, the natures of the data that must be recorded necessarily become more complex.

The ranking and sorting are additionally performed to improvise the algorithm. The advantage of using the technique in the proposed work restructures the process with KNN.

4.3 GENETIC ALGORITHM PROCESS IN NRGA

GA is a computational model to solve the problem of accurate data classification. In addition to existing GA selection is performed using roulette wheel selection method. Then ranking is an essential process after selection. The sorting technique used is the Pareto dominance to generate the non dominated fronts. The algorithm is a potential solution to specific problem on a simple chromosome and recombination operators to structures so as to preserve critical information.

The proposed NRGA is an optimization and search technique, based on the principles of genetics and natural selection. It allows a population composed of many individuals to evolve under specified selection
rules towards a state that maximizes the “fitness” (i.e., minimizes the cost function).

4.4 STEPS IN NRGA

NRGA is a multi-objective evolutionary algorithm that uses non-dominated sorting and a crowded-comparison approach to find a set of evenly distributed solutions to a multi-objective optimization problem.

**Sorting:** The sorting procedure, all non-dominated solutions are ranked 1 and are temporarily removed from the population. The next set of non-dominated solutions in the population is then defined and ranked 2. The procedure is continued until all the solutions are ranked.

**Initial population:** The first step in the implementation of any GA is to generate an initial population. Then the random population is generated based in the initial population. In most cases the initial population is generated randomly.

**Evaluation:** After creating an initial population each string is then evaluated. The genotypes are used in the reproduction operations whereas the values of the objective functions $ob \in OB$ are computed on basis of the phenotypes in the problem space ‘RP’ which is obtained via the Genotype-Phenotype Mapping (GPM). The value of objective function is represented by $ob$ and $OB$. The objective value of the proposed algorithm is based on 1-NN for the initial population size.
Figure 4.1 Improved cycles of genetic algorithm

Figure 4.1 shows the improved cycles involved in the process of GA. The improved cycles are ranked based roulette wheel selection and recombination mutation. An execution of a GA starts with a population of characteristically arbitrary chromosomes. These arrangements are then estimated and allocated for reproductive opportunities in such a manner that those chromosomes signify a superior resolution to the objective problem. They are agreed further likelihood chances to “reproduce” than those chromosomes which supply worse solutions. The integrity of a result is normally defined with respect to the existing population.
Fitness Assignment: After the evaluation phase, a fitness value is assigned based on the values of the objective function obtained during the evaluation phase. The average evaluation of all the strings in the population is performed with ranking and sorting.

Fitness is assigned based on a string’s rank in the population from microarray data GE sampling with roulette wheel selection. It is helpful to view the execution of the GA as a two stage process namely ranking and sorting. It starts with the current population. Selection is done for the current population to create an intermediate population. Then recombination and mutation are applied towards the intermediate population to create the next population. The process of moving from the current population to the next population comprises of one generation in the execution of a GA. The crowding distance is calculated before the next generation of ranking, recombination and mutation.

**Selection:** The application of the fitness criterion to choose which individuals from a population will go on with selection and mutation process for reproduction process. Finally rank level is estimated based on the Pareto sort.

**Reproduction:** The process involves a new generation of population is formed by selecting the fittest individuals in the current population. It is the survival of the fittest mechanism for microarray genes. Strings selected for reproduction are copied and entered to crossover process.

**Crossover:** The process is the creation of one or more offspring’s from the parents selected in the pairing process for microarray data. The current members of the population limit the genetic population for mutation process. It involves parents that produce two offspring’s and new offspring’s may replace the weaker individuals in the population. With the cross over operation, GA is able to acquire more information with the generated
individuals and the search space is thus extended. Hence, performing mutation is simple.

**Mutation:** Random mutations alter a certain percentage of the bits in the list of chromosomes.

The notion of evaluation and fitness are used interchangeably. However, it is useful to distinguish between the evaluation function and the fitness function used by a GA. The fitness function transforms that measured performance into an allocation of reproductive opportunities. The evaluation in a string representing a parameters set is independent of any other string. The fitness of that string, however, is always defined with respect to other members of the current population.

**Applications of non-dominated genetic algorithm**

The proposed approach uses the NRGA for the optimization purpose. The main advantages of using NRGA are that it converges very significantly than GA. Moreover, it provides rank based fitness function and it is quicker than GA for accurate cancer classification.

**Overview of feature selection using Information Gain**

Data sets for analysis may contain hundreds of attributes, many of which may be irrelevant to the mining task, or redundant. Although it may be possible for a domain expert to pick out some of the useful attributes, this can be a difficult and time consuming task, especially when the behaviour of the data is not well known. Leaving out relevant attributes or keeping irrelevant attributes may be detrimental, causing confusion for mining algorithm employed. Thus the dimensionality reduction reduces the data size by removing such attributes from it. The method called attribute subset selection
is applied to reduce the data size. The goal of attribute subset selection is to find a minimum set of attributes such that the resulting probability distribution of the data classes is as close as possible to the original distribution obtained using all attributes. Mining on a reduced set of attributes has an additional benefit. It reduces the number of attributes appearing in the discovered patterns, helping to make the patterns easier to understand.

**Discernibility matrix**

Let $(U, A)$ be an information gain system, where $U$ is a non empty finite set of objects and $A$ is a non empty set of attributes. $M$ be a set of $\{T\}$ called the discernibility matrix of $(U, A)$, such that $T = \{a \in A$, where $C[I] \neq C[J] \} I,J=1,2,\ldots,n$. The physical meaning of the matrix element $M(x, y)$ is that objects $x$ and $y$ can be distinguished by any attribute in $M(x, y)$. The pair $(x, y)$ can be discerned if $M(x, y) \neq 0$. A discernibility matrix $M$ is symmetric, i.e., $M(x, y) = M(y, x)$, and $M(x, x) = 0$. Therefore, it is sufficient to consider only the lower triangle or the upper triangle of the matrix.

**Discernibility Function:** For an information system $(U,A)$, s discernibility function $F$ is a boolean function of m Boolean variables $(a_1, a_2, \ldots, a_m)$ corresponding to the attributes $(a_1, a_2, \ldots, a_m)$ respectively, and defined as follows: $F(a_1, a_2, \ldots, a_m) = T_1, T_2, \ldots, T_m$ where $a_i \in T$.

**Absorptive law:** Let $G$ and $H$ be two logical formulas, $G \ast \wedge (G \vee \wedge H) = G \vee \wedge (G \wedge \ast H) = G$. Use distributive law(Multiplication) and absorptive law to transform the discernibility function into a Disjunctive Normal Form(DNF), which is just the reduct space of the given information system. Every clause in this DNF is a reduct.
Algorithm of IG for Feature Selection

Step 1: Compute discernibility matrix for the selected dataset.

By using \( M[I,J] = \{ a \in A, \text{where } C[I] \neq C[J] \text{ and } D[I] \neq D[J] \} \), \( I,J = 1,2,\ldots,n \)

Where \( C \) are conditional attributes and \( D \) is a decision attribute. This discernibility matrix \( M \) is symmetric. Where \( M[x,y] = M[y,x] \) and \( M[x,x] = 0 \). Therefore, it is sufficient to consider only the lower triangle or the upper triangle of the matrix.

Step 2: Compute the discernibility function for the discernibility matrix \( M[x,y] \) by using Equation (4.1)

\[
F(x) = \wedge \{ \wedge M[x,y] / x,y \in U; M[x,y] \neq 0 \} \quad (4.1)
\]

Step 3: Select the attribute, which belongs to the large number of conjunctive sets, numbering at least two, and apply the expansion law.

Step 4: Repeat steps 1 to 3 until the expansion law cannot be applied for each component.

Step 5: Substitute all strongly equivalent classes for their corresponding attributes.

Step 6: Calculate the Information gain for the simplified discernibility function contained attributes by using

The information gain is defined as given in Equation (4.2) which derives Equation (4.3) and Equation (4.4).

\[
Gain(S_j) = E(P_i) - E(S_j) \quad (4.2)
\]
Where

\[ E(P) = \sum_{i=1}^{n} P_i \log_2 P_i \]  \hspace{1cm} (4.3)

\[ \sum_{i=1}^{n} P_i \log_2 P_i = -\frac{p_1}{p} \log_2 \frac{p_1}{p} - \frac{p_2}{p} \log_2 \frac{p_2}{p} \ldots \frac{p_n}{p} \log_2 \frac{p_n}{p} \] \hspace{1cm} (4.4)

Where \( P_i \) is the ratio of conditional attribute P in dataset. When \( S_j \) has \( |S_j| \) kinds of attribute values and condition attribute \( P_i \) partitions set P using attribute \( S_j \), the value of information \( E(S_j) \) is defined as given in Equation (4.5).

\[ E(S_j) = \sum_{i=1}^{S_j} I_j * E(Y_j) \] \hspace{1cm} (4.5)

\[ E(Y_j) = \text{Set of instances} \]

\[ I_j = \text{the ratio of decision attribute in dataset} \]

**Step 7:** Choose the highest Gain value and add it to the reduction set, and remove the attribute from the discernibility function.

Goto step 6 until the discernibility function reaches null set.

### 4.5 NRGA TECHNIQUE

The IG for the entropy calculation is carried on to perform the initial population. It supports NRGA with selection approach. Further GA technique with ranking and sorting enhances the GA process. But, in opposition to the Nondominated Sorting Genetic Algorithm (NSGA-II), the difference between the NRGA and the NSGA-II is their different in selection approach. In NRGA, as an alternative of binary contest selection, roulette wheel selection is used. In that algorithm, a fitness value equal to its rank in the population is assigned to each individual.
Figure 4.2 describes the flowchart of the NRGA approach. The following characteristics are observed.

Figure 4.2 NRGA Approach

- Calculate the entropy values for each of the n genes
- Calculate the information gain entropy values for the n genes
- Initialize the population P
- Generate Random population of size N
- Evaluate the Fitness function based on equation 4.13
- Stop

Flowchart:

1. Calculate the entropy values for each of the n genes
2. Calculate the information gain entropy values for the n genes
3. Initialize the population P
4. Generate Random population of size N
5. Evaluate the Fitness function based on equation 4.13
6. Stop

Criteria:

Output

- Assign rank based on pareto –dominance sort
- Generate sets of non-dominated fronts
- Calculate crowding distance between each front
- Select the members from the least dominated N solution
- Generate child population based on Tournament Selection- Mutation & recombination

Stopping Criteria

- YES
- NO

Figure 4.2 NRGA Approach
Initially, arrange the population corresponding to fast non-domination sorting and select the better solutions from the first ranked population. Then, in the favour of crowding distance condition, individuals of each front are ranked. At present, two tiers ranked based roulette wheel selection are used (one tier to select the front and the other to select solution from the front) in Equation (4.6).

**Ranking**

In this selection operation all individuals are sorted by increasing values of fitness. Each individual is then assigned a probability, $p_i$, of being selected from some prior probability distribution.

The front probability obtained as

$$P_i = \frac{2 \times \text{rank}_i}{N_F \times (N_F + 1)} \forall, i = 1, \ldots, N_F$$

(4.6)

where, $N_F$ indicates the number of fronts. In the equation, it is understandable that a front with highest rank has the highest probability to be selected. So the probability of individual fronts based on their crowding distance measure is calculated as follows:

$$P_{ij} = \frac{2 \times \text{rank}_{ij}}{M_i \times (M_i + 1)} \forall, i = 1, \ldots, N_F, \forall, i = 1, \ldots, M_i$$

(4.7)

where, $M_i$ denotes the number of individuals in the front $i$. From the equation 4.7, it is observed that individuals with more crowding distance have more selection probability. The diversity between non-dominated solutions is also measured. After that, roulette wheel selection is applied to the two
random numbers in intervals [0, 1] correspondingly; the process is continual until the preferred number of individuals has been chosen which is clearly explained.

In conjunction with convergence to the Pareto-optimal set, it is preferred that the system preserves a good quality spread of solutions in the acquired set of solutions as discussed by Rajapakse & Mundra (2013). In NSGA-II the crowded-comparison approach is unaccompanied with the crowded-comparison operator.

For preserving diversity between population members this approach do not need any of the user defined component. In addition to that, the recommended method has an improved computational complexity. For more information about both the crowding-comparison approach and operator a detailed study is made. NRGA sustains the diversity by ranking the solutions in each non-dominated Pareto-front by means of their crowding distance.

After calculating the offspring’s fitness, parents and offspring struggle for endurance as Pareto dominance is applied to the combined population of parents and offspring. Afterward the least dominated N solutions stay alive to make the population of the next generation.

The crowding distance value of a solution gives an approximation of the density of solutions nearby that solution.
Figure 4.3 shows the computation of the crowding distance of point $i$ which is an estimation of the size of the largest cuboid enclosing $i$ without including any other point.

Crowding distance is estimated by the initial arrangement of the position of solutions in rising objective function values. The crowding distance value of its two adjacent solutions is determined for the particular solution. The boundary solutions which have the lowest and highest objective function values are given an infinite crowding distance values so that they are always selected. This process is done for each objective function. The final crowding distance value of a solution is computed by adding the entire individual crowding distance values in each objective function.

Pseudocode of crowding distance

1. Get the number of nondominated solutions in the external repository

$$ n = | S | $$

2. Initialize distance
FOR i=0 TO MAX

S[i].distance = 0

3. Compute the crowding distance of each solution

For each objective m

Sort using each objective value

4. $S = \text{sort}(S, m)$

For i=1 to (n-1)

$S[i].distance = S[i].distance + (S[i+1].m - S[i-1].m)$

Set the maximum distance to the boundary points so that they are always selected

$S[0].distance = S[n].distance = \text{maximum distance}$

Therefore, NRGA clustering technique has been utilized for the feature selection of microarray genes.

4.6 PROPOSED METHOD OF INFORMATION GAIN WITH NRGA FOR FEATURE SELECTION

Clustering is the assignment of combining a set of objects in such a way that objects in the identical group (called cluster) are alike (in some sense or another) to each other than to those in supplementary groups (clusters). The clustering method based on the IG with NRGA for feature selection is proposed here.
Figure 4.4 Overall methodology of feature selection

Figure 4.4 depicts the overall methodology of the feature selection process using IG. The data sets used in the feature selection process contain IG KNN blocks and the proposed IG NRGA KNN blocks in detailed manner.

**Overall process of feature selection using IG**

1. Given a set of examples $S$, and the number of different classes $c$, then the entropy of $S$ relative to this class--wise classification is calculated.
2. Then it is necessary to find the threshold, \( k \), here 0.

3. Find the greatest IG for \( A_k \). And this greatest gain value can represent the IG value, \( Gain(S, A) \) of gene \( A \).

4. Find IG KNN
   a. Calculate Entropy
   b. Set threshold to 0,
   c. If a feature has a greater IG value than the threshold, the feature is chosen; or else, it is not selected
   d. Calculate IG
   e. For \( k=1,2,..5 \), calculate mean, minimum distance
   f. Find minimum distance and sum of distance and time
   g. Calculate classification accuracy

5. Find IG GA KNN
   a. Follow steps (4a-4d)
   b. Create Initial population,
   c. Calculate fitness as in crossover and mutation-selected features (max value)
   d. Follow steps 4e-4g
   e. test 50% for with random permutation
   f. Calculate classification accuracy

6. Find NRGA KNN
   a. Follow steps (4a-4d)
   b. Create Initial population
   c. Generate Random Population
   d. Evaluate fitness function from equation 4.13
   e. Assign Rank based on pareto dominance sort
   f. Generate set of non-dominance fronts
   g. Calculate crowding distance
h. Select members from last dominated solution and rank
i. Tournament selection, crossover mutation and recombination
j. Create child population
k. Continue till stopping criteria
l. Test 50% for with random permutation
m. Calculate classification accuracy

The algorithm demonstrates the overall process of feature selection using IG. Various dataset of GE profiles like brain tumor, lung cancer and prostate tumor is taken. The required data is trained, loaded and processed to get the size of the data.

Three different techniques based on IG are carried out to select the features of the genes. Initially, IG KNN is processed by calculating the value of IG. Then the threshold is set as 0.783, if value greater than threshold features unselected, the length is displayed.

After that, the IG GA KNN is to be initialized. Apply IG GA KNN to the GE profiles. Then calculate the maximum value of population, fitness, crossover and mutation-selected features. Test 50% for with random permutation by setting threshold value. The feature selected and length of the genes is displayed.

Finally IG NRGA KNN is applied to the selected features of the genes. The results of IG, IG GA and NRGA are displayed.

**4.6.1 Information gain**

IG used in feature selection comprises of a filter approach. The idea behind IG is to select features that reveal the most information about the classes. Each feature has its own IG value which determines whether the
feature is to be selected or not. A threshold value is used for checking the features; if a feature has a greater IG value than the threshold, the feature is chosen; or else, it is not selected. Clustering is then done by learning the parameters of these models and the associated probabilities.

Let \( S \) be the set of \( n \) instances and \( C \) be the set of \( k \) classes. \( P(C_i, S) \) is the fraction in ‘\( S \)’ that has class \( C_i \). Then, the expected information from the class membership is given by Equation (4.8)

\[
I(S) = - \sum_{i=1}^{k} P(C_i, S) \times \log(P(C_i, S))
\] (4.8)

If a particular attribute \( A \) has \( v \) distinct values, the expected information is obtained by the decision tree in which \( A \) is the root and the weighted sum of expected information of the subsets of \( A \) is based on the distinct values. Let \( S_i \) be the set of instances and \( A_i \) the value of attribute \( A \):

\[
I_A(S) = - \sum_{i=1}^{v} \frac{|S_i|}{|S|} \times I(S_i)
\] (4.9)

Then, the difference between \( I(s) \) and \( I_A(S) \) provides the information gained by partitioning \( S \) according to the test \( A \) in Equation (4.9)

\[
Gain(A) = I(S) - I_A(S)
\] (4.10)

A higher IG will result in a higher likelihood of obtaining pure classes in a target class of Equation (4.10).

After measuring the IG values for all features, a threshold for the results was recognized. Since the results show that most IG values are zero after the computation process, not many features have an influence on the
category in a data set, representing that these features are irrelevant for classification. Threshold was 0 for most of the data sets. If the IG value of the feature was higher than the threshold, the feature was selected; if not, the feature was not selected according to the clustering technique.

4.6.1.1 **Entropy calculation using Information Gain**

The purpose of these techniques is to discard irrelevant or redundant features from a given feature vector. The following attribute evaluations are used: IG, gain ratio, symmetrical uncertainty, relief-F, one-R and chi-squared.

Entropy is a commonly used in the information theory measure, which characterizes the purity of an arbitrary collection of examples. It is in the foundation of the IG attribute ranking methods. The entropy measure is considered as a measure of system's unpredictability. The entropy of Y is given in Equation (4.11).

\[
H(Y) = -\sum_{y \in Y} p(y) \log_2(p(y)) \tag{4.11}
\]

where \( p(y) \) is the marginal probability density function for the random variable Y. If the observed values of Y in the training data set S are partitioned according to the values of a second feature X, and the entropy of Y with respect to the partitions induced by X is less than the entropy of Y prior to partitioning, then there is a relationship between features Y and X. Then the entropy of Y after observing X is given in Equation (4.12):

\[
H(Y \mid X) = -\sum_{x \in X} p(x) \sum_{y \in Y} p(y \mid x) \log_2(p(y \mid x)) \tag{4.12}
\]

where \( p(y \mid x) \) is the conditional probability of y given x.
Given the entropy as a criterion of impurity in a training set S, can define a measure reflecting additional information about Y provided by X that represents the amount by which the entropy of Y decreases. This measure is known as IG. It is given by Equation (4.13).

\[
IG = H(Y) - H(Y/X) = H(X) - H(X/Y)
\] (4.13)

IG is a symmetrical measure. The information gained about Y after observing X is equal to the information gained about X after observing Y. A weakness of the IG criterion is that it is biased in favor of features with more values even when they are not more informative.

4.6.2 Non Dominated Ranked Genetic Algorithm (NDRGA) method in proposed algorithms

The proposed approach uses the NDRGA for the optimization purpose. The main advantage is that it converges very significantly than GA. Moreover, it provides rank based fitness function and it is quicker than GA. After the IG phase, a fitness value is assigned based on the values of the objective function obtained during the previous phase. Like IG, it is also rank based method to improve the classification accuracy of genes.

4.6.3 KNN

The KNN technique is a supervised learning algorithm where the outcome of a new query occurrence is splitted based on the more number of KNN categories. The benefit of the KNN method is its simplicity, reliability and easy execution. In the investigation, the feature subset was calculated by the Leave-One-Out Cross-Validation of one Nearest Neighbour (1-NN). Neighbours are measured by means of their Euclidean distance. The 1-NN
classifier does not need any user-specified parameters and the classification results are carrying out by self-governing process.

4.6.4 NRGA algorithm with information gain and KNN (Gene Selection through Hybrid Approach)

In this thesis, a filter method (information gain, IG) and a wrapper method (non-dominated ranking genetic algorithm, NRGA) is proposed for feature selection in microarray data sets. IG was used to select important feature subsets (genes) from all features in the gene expression data, and a NRGA was employed for actual feature selection. The K-nearest neighbor (K-NN) method with Leave-One-out Cross-Validation (LOoCV) served as an evaluator of the IG-NRGA.

In the first-stage, IG is used to calculate the information gain values for each gene in the microarray data sets. Each gene obtains a value, which is regarded as a score, and the score can represent the importance of each gene. However, in this study, the disparities among the genes of the information gain values are not minded; just pay attention to the order of genes. After calculating the information gain values, the genes are sorted according to the information gain values. Some studies used a threshold for selecting how many genes. The threshold was zero for the data sets. If the information gain value of the feature was higher than the threshold, the feature was selected; if not, the feature was not selected.

Here it should be noticed that, generally step 6 of IG algorithm is only used to calculate the IG value for features which take on a discrete set of values. While since the gene expression values are continuous, it is not proper to simply apply IG method to the genes. Otherwise, new discrete valued
features should be defined that partition the continuous gene expression value into a discrete of intervals. In particular, for a gene \( A \), a new boolean feature \( A_k \) is created that is true if \( A < k \) and false otherwise, so that step 6 of IG algorithm is applied to \( A_k \). Then the threshold, \( k \), that produces the greatest IG for \( A_k \) and this greatest gain value represents the IG value, \( Gain(S, A) \) of gene \( A \). For this purpose, firstly the examples according to the gene expression values of \( A \) are sorted. Then by identifying adjacent examples that differ in their class label, a set of candidate thresholds are generated midway between the corresponding values of \( A \). It can be shown that the value of \( k \) that maximizes the IG value must always lie at such a boundary. So through calculating the IG value associated with each threshold, the greatest gain value can be found, which acts as the IG value of gene \( A \). The method which shown is used to calculate the IG value of each gene in two raw datasets. Then the genes are ranked in terms of their IG values, and the top genes (i.e., those with top IG values) are selected as the result.

The second stage is the selection of optimized features using NRGA. The third and final stage obtains the classification result. NRGA selects an optimal feature subset from a large number of feature sets. NRGA initializes the individual of population according to the comparative list from the first stage. Some subsets are selected by NRGA algorithm to be combined with KNN. Generally, the NRGA can find a sub-optimal subset from characteristic space. The final process obtains feature selection and classification result. NRGA selects an optimal feature subset from a large number of feature sets as given in Figure 4.4.a.
Figure 4.4.a Stages of Hybrid Approach in gene selection

In the study, the KNN and NRGA models are combined to select relevant genes. In the first-stage, IG, a filter method, was used to select informative genes. Initially, calculate the IG values for eleven GE data sets by Weka. IG values were calculated for each gene in the microarray data sets by IG and then the features were sorted in accordance with their IG values. A feature with a higher IG value indicates higher discrimination of the feature compared to other categories and means that the feature contains gene information useful for classification.
In the following example, GE data sets contain nine genes (features) which can be represented by F1, F2, F3, F4, F5, F6, F7, F8, and F9. After the application of IG, the nine IG scores were: F1 = 0, F2 = 0.4, F3 = 0, F4 = 0.9, F5 = 0, F6 = 1.2, F7 = 0.6, F8 = 0.5, F9 = 0. Since most of the scores were 0, so use 0 as the threshold value.

The five values that were above the threshold value (F2, F4, F6, F7, and F8) were then used to continue implementing the feature selection process in the second-stage. In the second stage the NRGA algorithm is introduced to increase the CA and searching abilities.

4.6.4.1 Chromosome Encoding and Initialization

Initially, a random parent population $P_o$ is created. The population is sorted based on non-domination. Each solution is assigned a rank or fitness equal to its non-domination level (1 is the best level, 2 is the next best level and so on). Thus, minimization of fitness is assumed. The $i^{th}$ string in the population is selected with a probability proportional to $F_i$. Since the population size is usually kept fixed in a simple GA, the sum of the probability of each string being selected must be one. Therefore, the probability for selecting the $i^{th}$ string is

$$P_i = \frac{F_i}{\sum_{i=1}^{n} F_i}$$  \hspace{1cm} (4.14)

Where $P_i$ the probability, $n$ is is the population size and $F_i$ is random variables.

At first, a random parent population P is formed. The random values for $F_i$ is chosen in the way that the selected random value must be within the limit specified in Equation (4.14).
4.6.4.2 Sorting (Non–dominated) and Ranking

To determine the range of potential population sizes, in the first run a small population size was used and the population size was doubled with each successive run. The percentage change in number of non-dominated individuals for two successive run is calculated for each successive run. The population size increase is no longer done when the percentage change in number of non-dominated individual fell below a pre-specified value (Equation 4.15)

\[
\Delta_{nd} < 100 \left| \frac{I_n - I_{n-1}}{I_{n-1}} \right| \quad (4.15)
\]

where \( \Delta_{nd} \) is the pre-specified percentage. \( I_n \) and \( I_{n-1} \) are the number of non-dominated individuals in successive runs.

The non–-dominated sorting and ranking steps are given below:

➢ For each individual \( p \) in main population \( P \) do the following
  - Initialize \( S_p = \emptyset \). This set would contain all the individuals that are being dominated by \( p \).
  - Initialize \( n_p = 0 \). This would be the number of individuals that dominate \( p \).
  - For each individual \( q \) in \( P \)
    * If \( p \) dominated \( q \), then
      - Add \( q \) to the set \( S_p \) i.e. \( S_p = S_p \cup \{ q \} \)
    * Else if \( q \) dominates \( p \) then
      - Increment the domination counter for \( p \) i.e. \( n_p = n_p + 1 \)
  - If \( n_p = 0 \), i.e., no individuals dominate \( p \), then \( p \) belongs to the first front; Set rank of individual \( p \) to one, i.e., \( p_{\text{rank}} = 1 \).
Update the first front set by adding $p$ to front one, i.e., $F_1 = F_1 \cup \{p\}$

- This is carried out for all the individuals in main population $P$.
- Initialize the front counter to one $i = 1$
- The following is carried out while the $i^{th}$ front is nonempty i.e. $F_i \neq \emptyset$

$Q = \emptyset$. The set for storing the individuals for $(i + 1)$ th front.

For each individual $p$ in front $F_i$

For each individual $q$ in $S_p$ ($S_p$ is the set of individuals dominated by $p$).

- $n_q = n_q - 1$, decrement the domination count for individual $q$.
- if $n_q = 0$, then none of the individuals in the subsequent fronts would dominate $q$. Hence set $q_{\text{rank}} = i + 1$. Update the set $Q$ with individual $q$, i.e., $Q = Q \cup q$.

- Increment the front counter by one.
- Now the set $Q$ is the next front and hence $F_i = Q$.

After sorting all the population, the first front i.e., the population with rank 1 is only considered. Remaining fronts are discarded. This algorithm utilizes the information about the set that an individual dominate ($S_p$) and number of individuals that dominate the individual ($n_p$).

### 4.6.4.3 Crowding Distance

Once the non-dominated sort is complete, the crowding distance is assigned. Since the individuals are selected based on rank and crowding distance, all the individuals in the population are assigned a crowding distance value.
The crowding distance is calculated as below:

- For each front \( F_i \), \( n \) is the number of individuals.
  - Initialize the distance to be zero for all the individuals i.e. \( F_i \)
    \( (d_j) = 0 \), where \( j \) corresponds to the \( jth \) individual in front \( F_i \).
  - For each objective function \( m \)
    - Sort the individuals in front \( F_i \) based on objective \( m \), i.e., \( I = \text{sort} (F_i, m) \).
    - Assign infinite distance to boundary values for each individual in i.e. \( F_i \).

The basic idea behind the crowding distance is finding the Euclidean distance between each individual in a front based on their \( m \) objectives in the \( m \) dimensional hyper space.

To get an estimate of the density of solutions surrounding a particular solution, we calculate the average distance of two points on either side along each objective is calculated by Equation (4.16).

\[
I_{\text{crowding}} = \sqrt{\left((i - 1) - (i + 1)\right)^2}
\] (4.16)

Therefore for each objective function, the boundary solutions are assigned an infinite distance value. The overall crowding distance is the sum of individual distance values to each objective.

The crowded comparison operator (\( \preceq_n \)) guides the selection process at the various stages of the algorithm. Let us assume that every individual \( I \) in the population has two attributes:

1. Non–domination rank (\( i_{\text{rank}} \))
2. Crowding Distance (\( i_{\text{crowding}} \))
The partial order $\preceq_n$ is as given in Equation (4.17):

$$i \preceq_n j \text{ if } (i_{\text{rank}} < j_{\text{rank}}) \text{ or } ((i_{\text{rank}} = j_{\text{rank}}) \text{ and } (i_{\text{crowding}} > j_{\text{crowding}}))$$ \hspace{1cm} (4.17)

### 4.6.4.4 Selection and Mutation

Once the individuals are sorted based on non-domination and with crowding distance assigned, the selection is carried out using a crowded-comparison-operator. The comparison is carried out as below based on

1. Non-domination rank $p_{\text{rank}}$, i.e., individuals in front $F_i$ will have their rank as $p_{\text{rank}} = i$, and
2. Crowding distance $F_i(\delta_j)$.

Binary tournament selection, recombination, standard two point cross over and standard uniform mutation operators are used to create a child population $Q_0$ of size $N$. For generations $t = 1$, a combined population $R_t = P_t \cup Q_t$ is formed first, which is of size $2N$. The population $R_t$ is then sorted according to non-domination. The new parent population $P_{t+1}$ is formed by adding solutions from the first front until the size exceeds $N$. Thereafter, the solutions of the last accepted front are sorted according to $n$, and the first $N$ points are picked. This is how the population $P_{t+1}$ of size $N$ is constructed. This population of size $N$ is now used for selection, crossover, and mutation to create a new population $Q_{t+1}$ of size $N$. That is, between two solutions with differing non-domination ranks, the solution with the lower (better rank) is preferred. The 2-point crossover operator used, which chose two cutting points at random and alternately copied single segments out of each parent. If a mutation was present, either one of the offsprings was mutated, and its binary representation changed from 1 to 0, or from 0 to 1 after the crossover operator is applied. If the mutated chromosome was superior to both parents, it replaced the worst chromosome of the parents; otherwise, the most inferior chromosome in the population was replaced. The GA was configured to contain 30 populations and was run for 100 generations in each configuration.
The crossover and mutation rates were 0.8 and 0.1, respectively. The child population is then analyzed and ranked.

This offspring is sorted again based on non-domination and only the best individuals with ranking 1 are selected. The selection is based on rank and the crowding distance of the last front. The new generation is filled by each front subsequently until the population size exceeds the current population size. If by adding all the individuals in front \( F_j \) the population exceeds \( N \), then individuals in front \( F_j \) are selected based on their crowding distance in the descending order till the population size becomes \( N \). And hence the process repeats to generate the subsequent generations.

4.6.4.5 Fitness Evaluation

The fitness can be calculated as in given in Equation (4.18).

\[
\bar{F} = \sum_{j=1}^{m} A(x) + \frac{P}{N(x)}
\]  

(4.18)

where for chromosome \( x \), \( A(x) \) is the classification accuracy of multi–class classifier defined as ration of number of correctly classified samples to total number of test samples and \( m \) is the number of objective function.

\[ P=\frac{100}{(M \times \text{Number of test samples used in classifier})} \]

\( N(x) \) is the size of gene set (number of 1’s in chromosome \( x \)) used for classification. The value of \( P \) chosen in fitness function will take care that number of genes are not minimized at the cost of accuracy.

Using the fitness value \( F_i \) of all strings, the probability of selecting string \( p_i \) can be calculated. Thereafter, the above probability ‘\( p_i \)’ of each string being copied can be calculated by adding the individual probabilities from the top of the list. Thus, the bottom-most string in the population should
have a probability values from \( P_{i-1} \) to \( P \). The first string represents the values from zero to \( P_1 \). The probability of any string lies between 0 to 1. As proposed, the NRGA algorithm, whose effectiveness can be determined by using them as features in an KNN classifier maintains a population of predictors. Using both the maximum number of generations and the criteria of no improvement of maximum fitness value of the population the termination criteria is defined.

4.6.4.6 Pseudocode for NRGA algorithm

Initialize Population \( F_i \) where \( i = 1, \ldots, n \)

\[
\{
\text{Generate random populations of } F_i - \text{size } n \text{ }

\text{Evaluate population objective values } J \text{ based on 1-NN for } F_i
\}
\]

Assign rank (level) random Populations of \( F_i \) based on Pareto Dominance sort

\[
\{
\text{Ranked based roulette wheel selection }

\text{Recombination and mutation }
\}
\]

\( Q \in F_i \)

for \( i=1 \) to \( g \) do

for each member of the combined population (PUQ) do

\[
\{
\text{Assign rank (level) based on Pareto-sort}

\text{Generate sets of non-dominated fronts}

\text{Calculate the crowding distance between members of each front}
\}
\]
(elitist) Select the members of combined population based on least dominated
n solution \( t_1 \) make the population of the next generation. Ties are resolved by
taking less crowding distance

Create next generation

\{
  Ranked based Roulette wheel selection
  Recombination Mutation
\}

The features selected during the first-stage were used for feature
selection by the NRGA algorithm. The chromosome length represents the
number of the features. The bit value \{1\} represents a selected feature,
whereas the bit value \{0\} represents a non-selected feature. The predictive
accuracy of a 1-NN determined by the LOoCV method was used to measure
the fitness of an individual. For example, when a 9-dimensional data set (n =
9) is analyzed, any number of features smaller than n can be selected. When
the adaptive value is calculated, these five features in each data set represent
the data dimension and are evaluated by the 1-NN method. The fitness value
for 1-NN evolves according to the LOoCV method for all data sets.

In the LOoCV method, a single observation from the original
sample is selected as the validation data and the remaining observations as the
training data. It is repeated so that each observation in the sample is used once
as the validation data. Essentially the K-fold cross-validation is used, where K
is equal to the number of observations in the original sample.

NRGA algorithm was implemented. Initially, a Population of \( F_1 \) is
created. Random Populations of \( F_1 \) is then generated which is of size N. Then
the objective function value of J (J= Objective value) is evaluated. Rank is
assigned to the Population with the best objective values based on the Pareto
dominance sort. Then the selection process is carried out based on the ranked based roulette wheel selection. Then in the Reproduction Phase (RP), recombination and mutation is carried out. RP produces new set of population \( Q \in F_1, T_{11} & T_{31} \) which are the points in the s-plane. A combined Population (RPUQ) is generated. Rank is assigned to the Population with the best objective values based on the Pareto dominance sort. The members are selected from the combined population based on least dominated N solution (elitist). The new population of size N is used for selection. Now, two tiers ranked based roulette wheel selection is applied. One tier is to select the front and the other to select solution from the front. Here the solutions belonging to the best non-dominated set have the largest probabilities to be selected. Then, in the RP, crossover and mutation are applied to create a new population RP of size N.

The diversity between non-dominated results is established by the second tier of ranked dependent roulette wheel selection that ranks the results according to their crowding distance. The results with lesser crowding distance will have the higher probabilities. As solutions contend with their crowding distance, no extra niching attribute is needed. Even though the crowding distance is computed in the objective function space, it can also be obtained in the parameter space, if required.

4.7 EXPERIMENTAL RESULTS

Feature selection improves classification accuracy because only few features necessarily influence the performance. Selecting appropriate features attributes according to the hybrid model, improves the accuracy; on the other hand, selecting inappropriate features attributes compromises the accuracy. Hence, employing appropriate feature selection to select optimal features for a category results in higher accuracy.
The data sets in the study was downloaded from http://www.gems-system.org. They include brain tumor, lung cancer and prostate tumor samples. Table 4.1 shows the format of GE classification data.

### Table 4.1 Format of gene expression classification data

<table>
<thead>
<tr>
<th>Data set</th>
<th>Samples</th>
<th>Genes</th>
<th>Class</th>
<th>Diagnostic task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brain tumor</td>
<td>90</td>
<td>5920</td>
<td>5</td>
<td>5 human brain tumor</td>
</tr>
<tr>
<td>Lung Cancer</td>
<td>203</td>
<td>12600</td>
<td>5</td>
<td>4 lung cancer and normal tissues</td>
</tr>
<tr>
<td>Prostate Tumor</td>
<td>102</td>
<td>10509</td>
<td>2</td>
<td>Prostate tumor and normal tissue</td>
</tr>
</tbody>
</table>

The microarray data was obtained by the oligonucleotide technique, except in the case of SRBCT, which was obtained by continuous image analysis. Table 4.2 shows the selected gene feature number using the filter, wrapper and hybrid feature selection using kNN. It is a fundamental task for the proposed approach to select genes in order to diagnose cancer tissues. The robustness of minimally selected genes classifies accurate samples.

### Table 4.2 The selected gene feature number for the three microarray data sets using filter, wrapper and hybrid feature selection using kNN

<table>
<thead>
<tr>
<th>Data set</th>
<th>KNN No feature selection</th>
<th>IG</th>
<th>GA</th>
<th>NRGA</th>
<th>IG/GA</th>
<th>IG/NRGA</th>
<th>Percentage of gene selected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brain tumor</td>
<td>5920</td>
<td>1612</td>
<td>2090</td>
<td>996</td>
<td>954</td>
<td>104</td>
<td>3.9%</td>
</tr>
<tr>
<td>Lung Cancer</td>
<td>12600</td>
<td>9561</td>
<td>8733</td>
<td>5321</td>
<td>2101</td>
<td>1845</td>
<td>15.7%</td>
</tr>
<tr>
<td>Prostate Tumor</td>
<td>10509</td>
<td>2016</td>
<td>3153</td>
<td>1094</td>
<td>788</td>
<td>343</td>
<td>3.3%</td>
</tr>
</tbody>
</table>
4.7.1 Performance Evaluation

The performance metrics considered in the present research work are accuracy and time taken. Three dataset like brain tumor, lung cancer and prostrate tumor is considered for this evaluation.

The two parameters are evaluated for sets of iterations namely 10, 100, 500 and 1000 iterations. The results of the proposed NRGA KNN are compared with IG KNN and GA KNN approaches.

4.7.2 Accuracy Comparison

For the 100 iterations, the accuracy of the proposed system increases when compared with the 10 iterations and for 1000 iterations also accuracy is increased on comparing with 500 iterations. The detailed formulation is given in 4.7.4.1.

Table 4.3 Classification accuracy comparison - dataset

(a) 10 and 100 iterations

<table>
<thead>
<tr>
<th>S. No.</th>
<th>Data sets</th>
<th>Accuracy % for 10 iterations</th>
<th>Accuracy for 100 iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Dataset</td>
<td>IG KNN</td>
<td>IG GA KNN</td>
</tr>
<tr>
<td>1.</td>
<td>Brain tumor</td>
<td>78.00</td>
<td>77.0</td>
</tr>
<tr>
<td>2.</td>
<td>Lung cancer</td>
<td>70.15</td>
<td>68.4</td>
</tr>
<tr>
<td>3.</td>
<td>Prostate tumor</td>
<td>72.22</td>
<td>77.6</td>
</tr>
</tbody>
</table>
Table 4.3

(b) 500 and 1000 iterations

<table>
<thead>
<tr>
<th>S. No.</th>
<th>Data sets</th>
<th>Accuracy % for 500 iterations</th>
<th>Accuracy for 1000 iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>IG KNN</td>
<td>IG GA KNN</td>
</tr>
<tr>
<td>1.</td>
<td>Brain tumor</td>
<td>51</td>
<td>63</td>
</tr>
<tr>
<td>2.</td>
<td>Lung cancer</td>
<td>43</td>
<td>55</td>
</tr>
<tr>
<td>3.</td>
<td>Prostate tumor</td>
<td>58</td>
<td>65</td>
</tr>
</tbody>
</table>

Table 4.3 shows that the Classification Accuracy for the GE data for four sets of iterations namely 10, 100, 500 and 1000 iterations. Multi dataset containing large amount of genes and small fraction of selection are compared for classification diagnostic accuracy. It can be visibly seen that the proposed NRGA KNN algorithm is 87% accurate in brain tumor, 94% accurate in lung cancer and 97% accurate in prostate tumor than the other existing approaches like IG KNN and IG GA KNN. On comparing to the existing methods for four 10, 100, 500 and 1000 iterations proposed methods leads the CA. Because the proposed approach converges very significantly than other algorithm due to use of ranked based roulette wheel selection. Moreover, it provides rank based fitness function and it is quicker than GA.
Figure 4.5 Accuracy comparisons for 10 and 100 iterations
Figure 4.5 compares the proposed NRGA approach with existing techniques, IG and IG GA resulting in increased accuracy for 10 and 100 iterations.

Figure 4.6 is presented with 500 and 1000 iterations. It shows the (graphical view) accuracy comparison of the proposed NRGA KNN approach compared with the existing GA KNN approach.

Accuracy comparison of 500 iterations

Existing and proposed methods

- Brain tumor
- Lung cancer
- Prostate tumor

(a) 500 iterations

Figure 4.6 (Continued)
Figure 4.6 Accuracy comparisons for 500 and 1000 iterations

It is observed that the proposed approach provides better accuracy when compared with the existing approach.

Brain tumor dataset

Figure 4.7 shows the accuracy comparison of the proposed NRGA KNN approach with the existing approach. The performance is evaluated for 10, 100, 500 and 1000 iterations in brain tumor dataset.
Figure 4.7  Accuracy comparisons for brain tumor dataset with 10, 100, 500 and 1000 iterations
It can be seen that the proposed NRGA algorithm which is more accurate in selecting the features than the existing IG GA. It is experimented that the proposed NRGA approach provides significant results for 10, 100, 500 and 1000 iterations. When the number of iterations increases, the accuracy also increases and thus it results in better performance.

**Lung cancer dataset**

The crucial genes are accurately classified for microarray dataset. The performance measure on lung cancer dataset exhibits improved classification when comparing proposed NRGA KNN approach with the existing approach. The dimension of the dataset is reduced using genetic search. The performance is evaluated for 10, 100, 500 and 1000 iterations in lung cancer dataset.

![Lung cancer chart](image)

(a) 10 and 100 iterations

**Figure 4.8 (Continued)**
Figure 4.8 Accuracy comparisons for lung cancer dataset with 10, 100, 500 and 1000 iterations

Figure 4.8 obviously show that the proposed NRGA KNN approach outperforms the existing approach in terms of accuracy. The practical solution obtained from the graph proves that the proposed NRGA approach provides significant results for 10, 100, 500 and 1000 iterations. The average accuracy attained for 100 iterations is higher when compared with the average accuracy attained for 10 iterations and 1000 higher than 500.

Prostrate tumor

In order to obtain accurate diagnostic test, the proposed techniques imply accuracy comparison of the proposed NRGA KNN approach and existing GA KNN approach. The performance is evaluated for 10, 100, 500 and 1000 iterations in prostrate tumor dataset.
Figure 4.9 clearly shows that the proposed NRGA KNN approach improves the existing approach in terms of accuracy. It is experimented from the graph that the projected NRGA approach provides enhanced results for 1000, 500, 100 iterations when compared with 10 iterations.
It can be clearly seen that the proposed NRGA KNN for 10 iterations attains accuracy of about 86.8% for brain tumor, 77.4% for lung cancer and 79.4% for prostate tumor. When the iterations are increase to 100, NRGA KNN attains a result of 87.12% for brain tumor, 94.4% for lung cancer and 97.16% for prostate tumor. Hence, the CA shown attained by IG GA KNN is lesser when compared with the proposed NRGA KNN approach.

4.7.3 Classification Time Comparison

The Classification Time (CT) for the GE dataset of iterations namely 10, 100, 500 and 1000 iterations on average can be obviously seen that the proposed NRGA KNN algorithm consumes lesser time when compared to the other existing methods for 10, 100, 500 and 1000 iterations. It is a new method to classify gene expression on monitoring the run time via integration of computing networks.

Table 4.4 Classification time comparison - dataset

(a) 10 and 100 iterations

<table>
<thead>
<tr>
<th>S. No.</th>
<th>Data sets</th>
<th>Classification time (ms) for 10 iterations</th>
<th>Classification time (ms) for 100 iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>IG KNN</td>
<td>IG GA KNN</td>
</tr>
<tr>
<td>1.</td>
<td>Brain tumor</td>
<td>0.35</td>
<td>0.25</td>
</tr>
<tr>
<td>2.</td>
<td>Lung cancer</td>
<td>0.76</td>
<td>0.56</td>
</tr>
<tr>
<td>3.</td>
<td>Prostate tumor</td>
<td>0.48</td>
<td>0.38</td>
</tr>
</tbody>
</table>
Table 4.4 (Continued)

(b) 500 and 1000 iterations

<table>
<thead>
<tr>
<th>S. No.</th>
<th>Data sets</th>
<th>Classification time (ms) for 500 iterations</th>
<th>Classification time (ms) for 1000 iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>IG KNN</td>
<td>IG GA KNN</td>
</tr>
<tr>
<td>1.</td>
<td>Brain tumor</td>
<td>0.31</td>
<td>0.32</td>
</tr>
<tr>
<td>2.</td>
<td>Lung cancer</td>
<td>0.69</td>
<td>0.75</td>
</tr>
<tr>
<td>3.</td>
<td>Prostate tumor</td>
<td>0.35</td>
<td>0.43</td>
</tr>
</tbody>
</table>

Table 4.4 shows the time requirement for classification of 10, 100, 500 and 1000 iterations indicating reduced time for the proposed approach. The proposed method reduces 0.13 ms for brain tumor, 0.01 ms for lung cancer and 0.28 ms for prostate tumor for 10 iterations when compared to the existing IG GA. For 100 iterations, 0.09 ms for brain tumor, 0.2 ms for lung cancer and 0.05 ms for prostate tumor lesser than the existing method. For 500 iterations proposed method executed 0.07 ms for brain tumor, 0.01 ms for lung cancer and 0.08 ms for prostate tumor lesser than the existing IG GA. For 1000 iterations, 0.05 ms for brain tumor and 0.32 ms for lung cancer lesser than the existing method. The similar time 0.47 ms is taken for prostate tumor in 1000 iteration. It is observed that the proposed NRGA KNN approach provides better results than the other two existing approaches.
Figure 4.10 (Continued)

(a) 10 iterations

Classification time (ms) for 10 iterations

Existing and proposed methods

- Brain tumor
- Lung cancer
- Prostate tumor

Figure 4.10 Execution speed for 10 and 100 iterations

(b) 100 iterations

Classification time comparison (ms) for 100 iterations

Existing and proposed methods

- Brain tumor
- Lung cancer
- Prostate tumor
Figure 4.10 executes of 10 and 100 iterations indicating reduced time for the proposed approach. The proposed method reduces 0.13 ms for brain tumor, 0.01 ms for lung cancer and 0.28 ms for prostate tumor for 10 iterations when compared to the existing IG GA. For 100 iterations, 0.09 ms for brain tumor, 0.2 ms for lung cancer and 0.05 ms for prostate tumor lesser than the existing method.

![Classification time comparison (ms) for 500 iterations](chart.png)

**Existing and proposed methods**

- Brain tumor
- Lung cancer
- Prostate tumor

(a) **500 iterations**

*Figure 4.11 (Continued)*
Figure 4.11 also shows low rate of time for 500 and 1000 iterations in proposed NRGA KNN. For 500 iterations proposed method executed 0.07 ms for brain tumor, 0.01 ms for lung cancer and 0.08 ms for prostate tumor lesser than the existing IG GA. For 1000 iterations, 0.05 ms for brain tumor and 0.32 ms for lung cancer lesser than the existing method.

The similar time 0.47 ms is taken for prostate tumor in 1000 iteration. The dataset is minimized with service quality in time to speed up the development process.

It is observed that the proposed NRGA KNN approach provides better results than the other two existing approaches.
Brain tumor dataset, lung cancer dataset and prostrate tumor

The gene selection continues executing with minimum time for brain, lung and prostate cancer. The experimental execution performs better gene selection and the time comparison of the proposed NRGA KNN approach with the existing approach is illustrated. The performance is evaluated for 10, 100, 500 and 1000 iterations in brain tumor, lung cancer and prostate tumor dataset.

It can be seen that the proposed NRGA algorithm consumes 0.12 ms in brain tumor, 0.55 ms in lung cancer and 0.10 ms in prostate tumor for 10 iterations. Approximately NRGA is proposed with difference 20 ms lesser in brain tumor, 0.01 ms lesser in lung cancer and 0.28 ms lesser in prostate tumor. There is lesser execution time in selecting the features for proposed than the existing IG GA technique. The similar condition of improved less execution time in the proposed system exists.

Brain tumor

(a) 10 and 100 iterations

Figure 4.12 (Continued)
(b) 500 and 1000 iterations

(c) 10 and 100 iterations

Figure 4.12 (Continued)
Figure 4.12 (Continued)

(d) 500 and 1000 iterations

- **Lung cancer**

- **Prostate tumor**

(e) 10 and 100 iterations
(f) 500 and 1000 iterations

**Figure 4.12 Execution time comparisons in brain tumor, lung cancer and prostate tumor dataset for 10, 100, 500 and 1000 iterations**

From Figure 4.12 it is observed from the graph that the proposed NRGA approach provides significant results for 10, 100, 500 and 1000 iterations. When the number of iterations increases, the time taken also increases. Thus for 10 iterations, time taken is lesser but for 100 iterations, it is higher. The different sets of iterations are 10, 100, 500 and 1000. On comparing existing system with proposed system for each set of iteration proposed result takes the lead.

Hence, when the iteration increases accuracy also increases. The KNN method is served as an evaluator of the NRGA algorithm. For more than 10 iterations there exists a controversy with accuracy and time by applying NRGA without KNN. The existing method IG GA KNN process takes more time than NRGA KNN. NRGA algorithm sustains results with 100 iterations.
4.7.3.1 Time Complexity

The classification time depends on the time complexities. Table 4.5 provides the complexity analysis of the KNN, IG/KNN, GA/KNN, IG-GA/KNN and IG-NRGA/KNN approach.

**Table 4.5 Time dependency of different methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>KNN</td>
<td>$O(nm)$  &lt;br&gt; N represents the dimension of the data sets and m represents the number of samples</td>
</tr>
<tr>
<td>IG/KNN</td>
<td>$O(n \log n + nm)$  &lt;br&gt; n represents the number of samples, m represents the dimension of the data sets.</td>
</tr>
<tr>
<td>GA/KNN</td>
<td>$O(nmpg)$  &lt;br&gt; n represents the number of samples, m represents the dimension of the data sets, p represents the population size and g represents the number of generations.</td>
</tr>
<tr>
<td>NRGA/KNN</td>
<td>$O(MI(N-M))$  &lt;br&gt; where I is the no of set of non-dominated solution sets, M is the no of objective functions and N is the total no of solution sets. This is the worst case complexity.  &lt;br&gt; $O(M(N \log N)$ as this is the complexity for sorting M lists. This is the best case complexity.</td>
</tr>
<tr>
<td>IG-GA/KNN</td>
<td>$O(n \log n + nmpg)$  &lt;br&gt; n represents the number of samples, m represents the dimension of the data sets, p represents the population size and g represents the number of generations</td>
</tr>
<tr>
<td>IG-NRGA/KNN</td>
<td>$O(MI(N-M)) + O(nm)$  &lt;br&gt; where I is the no of set of Non-Dominated solution sets, M is the no of objective functions and N is the total no of solution sets. This is the worst case complexity.  &lt;br&gt; $O(M(N \log N)+ O(nm)$ as this is the complexity for sorting M lists. This is the best case complexity.</td>
</tr>
</tbody>
</table>
The time complexities above clearly show that the computation time needed for IG-NRGA/ KNN is less than other feature selection models with the KNN classifier.

4.7.4 Model Estimation and other evaluation criteria on classifiers

In order to compare the performance across different gene selection methods and classifiers, several evaluation criteria are introduced. These tools are for evaluating both the qualities of gene selection and classification methods. The first one is accuracy, which is the most common tool for evaluating whether the samples in the two classes are well separated or not. Another evaluation criterion relates to testing if the selected genes are stable. Other criteria include measures in relation to the confusion matrix, such as sensitivity (SEN), specificity (SPEC) and Matthew’s Correlation Coefficient (MCC). Each of these will be described in details in the following sessions.

4.7.4.1 Accuracy

Accuracy is a common criterion used for comparing the performance of different gene selection algorithms. The use of accuracy can help to determine whether the gene sets have the ability to discriminate the samples in the data well. There are two types of accuracies: one is the classification accuracy (also known as training accuracy), which is calculated from the training samples only. The other is the prediction accuracy (also called testing accuracy, or denoted as Leave-one-out Cross-validation accuracy, Leave One out cross validation LOoCV in the later parts of the thesis) which is obtained from testing samples only.

4.7.4.2 Measures in Relation to the Confusion Matrix

Although classification accuracy is a popular tool used for comparing performance of different classifiers, it is not suitable to be used in
data where the sample sizes of the two classes differ much. More appropriate evaluation criteria are those derived from the confusion matrix. Let us consider a binary classification problem, in which the outcomes are labeled either as positive (p) or negative (n) class. There are four possible outcomes from a binary classifier. If the outcome from a prediction is p and the actual value is also p, then it is called a true positive TP; however if the actual value is n then it is said to be a false positive FP. Conversely, a true negative TN has occurred when both the prediction outcome and the actual value are n, and false negative FN when the prediction outcome is n while the actual value is p. The evaluation of a classifier is most often based on its predictive accuracy. A confusion matrix describes the number of correctly and incorrectly predicted examples by the classification model. Table 4.6 shows the confusion matrix in a 2×2 contingency table or a confusion matrix.

Table 4.6 A confusion matrix summarizes all possible outcomes in a binary classification problem.

<table>
<thead>
<tr>
<th>Predicted Class</th>
<th>Class =0</th>
<th>Class =1</th>
</tr>
</thead>
<tbody>
<tr>
<td>True Class Class =0</td>
<td>f_{00}</td>
<td>f_{01}</td>
</tr>
<tr>
<td>Class=1</td>
<td>f_{10}</td>
<td>f_{11}</td>
</tr>
</tbody>
</table>

In Table 4.5, each entry $f_{ij}$ denotes the number of examples whose true class label is $i$ and predicted class label is $j$. For example, $f_{11}$ is the number of examples from class 1 correctly predicted as class 1 and $f_{10}$ is the
number of examples from class 1 incorrectly predicted as class 0. The goodness of a classifier for a binary classification problem can be easily predicted from the confusion matrix. But for multiclass classification problems, a generalize measure metric is needed to evaluate the performance of a classifier, such as accuracy, which is defined as follows in Equation (4.19):

\[
\text{Accuracy} = \frac{\text{Number of correctly predicted records}}{\text{Total number of predicted records}} \tag{4.19}
\]

For a binary classification problem, the definition of accuracy can be expressed as given in Equation (4.20):

\[
\text{Accuracy} = \frac{\text{Number of correctly predicted records}}{\text{Total number of predicted records}} = \frac{f_{00} + f_{11}}{f_{00} + f_{01} + f_{10} + f_{11}} \tag{4.20}
\]

Based on the definition of accuracy, using cross validation, the performance of a classifier is evaluated.

Common measures for comparing performance of different classifiers based on the confusion matrix are sensitivity and specificity. Sensitivity (SEN) is the probability of predicting a positive outcome when the true state is positive whereas specificity (SPEC) is the probability of predicting a negative outcome when the true state of a case is negative. While there is no perfect way of describing the confusion matrix of TP, FP, FN and TN by a single number, the Matthews Correlation Coefficient (MCC) is generally regarded as one of the best measures. Other measures, such as classification accuracy, are not useful when the sizes of the two classes are very different, which is a common phenomenon in microarray data.
MCC takes into account TP, FP, FN and TN so it is generally regarded as a balanced measure for use even if the classes are of very different sizes. It returns a value of between -1 and +1: +1 represents a perfect prediction, 0 an average random prediction and -1 an inverse prediction. The measures are summarized in relation to the confusion matrix discussed in this session by Equation (4.21) and Equation (4.22) that computes MCC as given in Equation (4.23):

\[
\text{Sensitivity} = \frac{TP}{FN + TP} \quad (4.21)
\]

\[
\text{Specificity} = \frac{TN}{FP + TN} \quad (4.22)
\]

\[
\text{MCC} = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \quad (4.23)
\]

The experiment starts by leaving out one sample in the data. Suppose the experimental data contains S samples. Using the S − 1 samples present in the dataset as the training data, the full set of genes G (total number of genes present in the data) is first filtered by a filter, IG to extract a subset of genes F that is much reduced in size compared to G. Each filter selects 200 genes in the experiment. The initial size of the final gene set to be zero, which means that a wrapper starts its search from an empty set. The genes in F are then selected in an incremental manner (forward selection) using a chosen wrapper, the Genetic Algorithm (GA) and Non-dominated Ranking Genetic algorithm (NRGA) with an aim to optimize the classification accuracy based on training data.

To start with, each gene in F is considered as a candidate for a single-gene classifier and the gene(s) giving the highest classification accuracy is/are identified and retained as the first gene in the set P. Then, the remaining genes in F are combined with the first gene in P to give a two-gene
classifier, and the second gene that together with the first selected gene yield the highest classification accuracy is identified. This process of adding genes to P is repeated until P contains a sufficient number of genes to achieve the best local optimum classification accuracy. The final gene set P is then evaluated using the left-out sample. This procedure was done iteratively by leaving out one sample each time (i.e. repeated S times). As a result, S sets of genes are produced. The classification accuracy is determined by counting the number of correctly predicted samples over S. The performance based on several criteria: LOoCV accuracy, Sensitivity (SEN), Specificity (SPEC) and MCC.

For each dataset, those models that output the best LOoCV accuracy are highlighted. It is concluded that for each data, different the hybrid models achieve the best performance, i.e. there is no single filter model that can achieve the best LOoCV accuracy (with respect to the data) across all the data that were used. The best model in terms of unbiased LOoCV accuracy is IG-NRGA +KNN, in which it can achieve the best ever accuracy for all three out of three datasets.

**Table 4.7 SEN values of all models for BRAIN, LUNG and Prostate respectively**

<table>
<thead>
<tr>
<th>Models</th>
<th>Brain</th>
<th>Lung</th>
<th>Prostate</th>
</tr>
</thead>
<tbody>
<tr>
<td>IG-KNN</td>
<td>0.78</td>
<td>0.84</td>
<td>0.76</td>
</tr>
<tr>
<td>GA-KNN</td>
<td>0.91</td>
<td>0.88</td>
<td>0.73</td>
</tr>
<tr>
<td>NRGA-KNN</td>
<td>0.88</td>
<td>1.00</td>
<td>0.78</td>
</tr>
<tr>
<td>IG-GA-KNN</td>
<td>0.96</td>
<td>0.89</td>
<td>0.82</td>
</tr>
<tr>
<td>Proposed IG-NRGA KNN</td>
<td>0.98</td>
<td>1.00</td>
<td>0.84</td>
</tr>
</tbody>
</table>
Table 4.8 SPEC values of all models for BRAIN, LUNG and Prostate respectively

<table>
<thead>
<tr>
<th>Models</th>
<th>Brain</th>
<th>Lung</th>
<th>Prostate</th>
</tr>
</thead>
<tbody>
<tr>
<td>IG-KNN</td>
<td>0.85</td>
<td>1.00</td>
<td>0.87</td>
</tr>
<tr>
<td>GA-KNN</td>
<td>0.96</td>
<td>0.90</td>
<td>0.81</td>
</tr>
<tr>
<td>NRGA-KNN</td>
<td>0.90</td>
<td>0.90</td>
<td>0.88</td>
</tr>
<tr>
<td>IG-GA-KNN</td>
<td>0.89</td>
<td>1.00</td>
<td>0.87</td>
</tr>
<tr>
<td>Proposed IG-NRGA KNN</td>
<td>0.94</td>
<td>1.00</td>
<td>0.88</td>
</tr>
</tbody>
</table>

SEN and SPEC values of Table 4.7 and Table 4.8 are highly sensitive to the sample sizes of the data. The problem of imbalance class size is most serious for the Lung dataset (203 samples versus 104 samples). For all microarray datasets included in our study, the positive class represents the class of larger sample size. The best SEN and SPEC values of LUNG are +1 (as the number of samples present in the positive class is much larger, samples from the larger class are more easily to be classified correctly). In order to provide a balanced measure for evaluating the quality of binary-class classification, MCC values are calculated. MCC of +1 indicates perfect prediction.

An interesting observation is that all hybrid models on LUNG achieve an MCC value of +1 which indicates that the SEN values of LUNG are actually not affected by the problem of imbalance class size. The perfect MCC value of LUNG shows the goodness of the proposed IG-NRGA-KNN model. Yet this is not the case for BRAIN as the best MCC value for BRAIN is only 0.91. Best MCC values for remaining datasets are as follows – LUNG: 1.00; and PROS: 0.73 as given in Table 4.9.
Table 4.9 MCC values of all models for BRAIN, LUNG and Prostate respectively

<table>
<thead>
<tr>
<th></th>
<th>Brain</th>
<th>Lung</th>
<th>Prostate</th>
</tr>
</thead>
<tbody>
<tr>
<td>IG-kNN</td>
<td>0.91</td>
<td>1.00</td>
<td>0.65</td>
</tr>
<tr>
<td>GA-kNN</td>
<td>0.88</td>
<td>0.92</td>
<td>0.73</td>
</tr>
<tr>
<td>NRGA-kNN</td>
<td>0.79</td>
<td>0.90</td>
<td>0.68</td>
</tr>
<tr>
<td>IG-GA-kNN</td>
<td>0.88</td>
<td>1.00</td>
<td>0.70</td>
</tr>
<tr>
<td>Proposed IG-NRGA KNN</td>
<td>0.91</td>
<td>1.00</td>
<td>0.73</td>
</tr>
</tbody>
</table>

The selected genes are not ‘biased’ towards any particular filters or wrappers. As a consequence, the generalizability of the selected genes with the model is much better. The number of genes \( m \) chosen by each filter should have an effect on the hybrid model performance because it determines the search space of the wrapper part and the final effectiveness of the gene selection.

As a consequence, the value of \( m \) should be carefully chosen.

- If \( m \) is too small, many potential useful genes are removed from the starting set of genes. No matter how powerful the classification model is, the accuracy obtained is not satisfactory as some useful genes are left out from the beginning of the experiment. Hence, the classification can be no better than what can be achieved from the set of genes captured by the filters.

- If \( m \) is too large, the computational time required is larger because each of the genes in the union set needs to combine with the previous selected gene set, and be evaluated by the hybrid model before deciding which gene(s) is(are) going to be
chosen in a particular iteration/level. To search for the best \( m \) value (say, among \( m = 50, 100, 150 \) and \( 200 \)) experiments were performed on all three microarray datasets using IG=NRGA-KNN model.

Thus the aim of evaluating a classifier is to ensure that it serves as a general model in which the input derived from training dataset can apply equally well on unseen test data from similar samples that are not included in the training set, i.e. the ability of generalization to new data. Although we can never exactly estimate the true generalization error of the model, proper evaluation can help to reduce the likelihood of overfitting of these classification models. The most straightforward way of obtaining unbiased estimates of a predictor’s is error rate.

4.8 SUMMARY

The chapter discusses on the proposed technique of the gene expression data. The chapter clearly explains about NRGA algorithm and it is used to perform feature selection based on clustering technique. The KNN method with LOoCV served as an evaluator of the NRGA fitness functions. Experimental results showed that NRGA simplified feature selection by clustering effectively reducing the total number of features needed, and obtained a higher accuracy compared to other feature selection methods in most cases. The difference of accuracy derived from the proposed strategy has a profound effect on overall result. Hence, compiling the information for NRGA with 1000 iterations allow 21% higher and accurate diagnosis. At the same time the execution speed for selecting features to diagnose cancer is 0.90 ms lesser. The ultimate importance of outcome is improved in its value without degrading the structure of proposed system and is very significant when compared with existing techniques.