CHAPTER – 5

OPTIMUM RETURN ON SHARE MARKET INVESTMENT THROUGH GENETIC ALGORITHMS
CHAPTER - 5

OPTIMUM RETURN ON SHARE MARKET INVESTMENT THROUGH
GENETIC ALGORITHMS

Abstract:

Predicting stock investments with traditional time series analysis and other statistical methods have proven to be difficult, since the vast amount of data and scripts are handled. Moreover, the data available in the Indian stock trading are not fixed and always uncertainty and ambiguity in nature. Therefore, the large number of researchers has proposed many new methods/models, which are based on the non-traditional tools like Fuzzy Logic, Neural Networks etc. The neural network is used to find the optimum number of iterations is required to train FIS rules. Again, for increasing the efficiency of the investing model, we trained these FIS rules by Neural Network (called ANFIS) and to reduce the scripts by a smaller number of scripts, so that one can manually picked the optimum rule. The disadvantage of this method is that the invested amount goes only to a limited number of scripts, it resulted that it will be automatically increased the risk. In this chapter, all the rules established from ANFIS are taken into account and genetic algorithm is used.

- To find the order (sequence of preference) in which one can invest the amount, if the investors want to invest their amount in a limited sub-
indexes and which is different from the approach given by the authors.[Abraham (2003), Alemdar (1998)].

- To find the sequence of preference together with the investing amount preference like high, medium, and low. (In these cases the investors have to invest their amount to the entire sub-indexes with the amount preference).

In this contest, two genetic algorithms are simultaneously executed, one will take care of arranging the sub-index preferences and the other will take care of amount of investment like high, medium, and low. This will help the investors to invest their amount, based on the preference given by genetic algorithm, to all type of sub-indexes. Another advantage of this chapter is to establishing the fitness function, which include the 'weight value of the sub-indexes' and the 'weight value of the risk'. Finally, the designed model is tested and implemented through BSE stock market index.

5.1 Introduction

Prediction of stocks is generally believed to be a very difficult task - it behaves like a random walk process and time varying. The obvious complexity of the problem paves the way for the importance of intelligent prediction paradigms. During the last decade, stocks and futures traders have come to rely upon various types of intelligent systems to make trading decisions [Abraham (2001, 2003), Datz et al.(2000), Droms (1989)]. Several intelligent systems have in recent years been developed for modeling expertise, decision support and complicated automation tasks [Leigh (2002a, 2002b, 2002c)].
Rapidly growing financial markets bring a wide range of specific problems to a new generation of practical investors responsible for the management of billions of dollars on behalf of banks, mutual funds and other financial institutions or private money holders. Quantitative methods become deadly important for the selection and further management of an appropriate investment portfolio in situations where the final choice has to be made out of hundreds of assets and other financial instruments presented on the national or international financial markets.

Selection of Index in a share market is a method of passive portfolio management. It attempts to match the performance of a theoretical portfolio, such as the BSE index, as closely as possible. This approach is used by fund managers or general investors when they do not feel confident enough of outperforming the market, and are content to follow the average performance.

Matching the performance of an index can be performed in two ways. The first is full selection, in which an investment is made in every sub-indexes of the BSE index, proportional to its market share. This achieves a perfect match, but incurs high initial transactions costs, and is difficult to rebalance when changes are made to the index, for example, the issue of a new set of shares. The other approach is partial selection, in which an investment is made in a small proportion of the shares, while attempting to match the performance of the entire index. This incurs lower initial transaction costs, and is easier to rebalance. It also allows the investor to constrain the choice of investment that is made, by
insisting on the inclusion or exclusion of some scripts / companies or by setting
the proportion of the capital that is to be invested in others. In this chapter, we
will give scope of full and partial investment strategy for those who have
selected a limited script to invest or to invest their money invariably to all
scripts, so that one can earn more return on investment.

Partial selection, however, introduces a sequence of performance, the
measure of the deviation of the chosen sub-indexes from the available BSE
index. To maximize the return on investment is the main aim of the fund
investors. This can be expressed as a fitness function in the first genetic
algorithm. The general frame work of genetic algorithms and the usage of
genetic algorithms in various engineering fields are demonstrated [ClaudioV.

This requires two pieces of data, the covariance matrix defining the
relationship between all of the sub-indexes in the index, and the weights of the
sub-indexes and its corresponding risk values [Ravichandran et al. 2005a]. The
covariance matrix shows the correlations that exist between sub-indexes in the
index. A large positive value occurs when the return from two companies follow
very similar trajectories. A small negative value occurs when two companies
follow roughly opposing trajectories. The weights are simply the normalized
returns for the index and the risk factor involved them. They are calculated by
taking the return for each sub-index, and dividing it by the sum of the returns for
all companies.
The problem, then, is to select a subset of all of the shares in an index which best matches the performance of the market as whole. The selection of subsets is algorithmically intractable, so stochastic search systems, such as genetic algorithms suggest themselves.

The genetic algorithm will be used to select subsets of shares, and sequence optimization will perform the index selection, evaluating the performance of each particular subset. This will be followed by a description of how certain operations define the power of genetic algorithms. The final sections discuss the testing, the results achieved, and what conclusions can be drawn from them.

In the next part of the work of this chapter is to find the optimization of FIS rule, that one can get the good return on investment. In this case, we consider the return on investment is based on inclusion of risk factor also. Since all the 11 types of scripts, as far as BSE index is concerned, are linguistically divided into three variables, namely high investment, medium investment and low investment. If possible, we can split the linguistic variables into further linguistic variables like low medium, high medium etc., which will increase the cost of computation time. When we apply FIS using mat lab 7.0, the system will get $3^{11}$ different rules. Out of these rules, the genetic algorithm is used to find the optimal investment rules. Section 5.2 deals with some basic operators used in genetic algorithm and finally this section will describe the importance of the fitness function. Section 5.2.1 explains that the performance of various operators
to the investment problem and will give the working procedure of the all the operators in one cycle. The complete evaluation of the genetic algorithm and its evaluation procedures are given in section 5.2.2 and finally, the conclusion is given in section 5.4.

5.2. Fuzzy Inference System

Fuzzy inference systems, which are also called fuzzy rule-based systems, or fuzzy models, are composed of 4 blocks (see Figure 5.1).

- **Fuzzifier**: Transforms the crisp inputs into fuzzy inputs by membership functions (MFs) that represent fuzzy sets of input vectors. In this work, singleton fuzzifier is assumed;

  \[ \mu_A(x) = 1, \text{ for } x = x' \text{ and } \mu_A(x) = 0 \text{ for } \]

  That is \( x \in U \) with \( x \neq x' \).

- **Rules**: Consists of fuzzy IF-THEN rules;

- **Inference**: Inference engine for fuzzy rules;

- **Defuzzifier**: Transforms the fuzzy output into crisp output. Defuzzification process requires the most computational complexity in FIS, and center-of-gravity or height defuzzification method is common.

The major component in an FIS is “Rules”, and Rules are expressed in the form of IF-THEN statements. Let \( U \) and \( V \) be universe of discourses for antecedent
and consequent of the rules, then the rule of if \( x \) is \( A \), then \( y \) is \( B \), where \( x \in U \), and \( y \in V \), represents a relation between \( A \) and \( B \), and extension to multiple rules and multiple antecedents can be easily done by specifying both composition and inference methods [John Holland.(1975)].

Figure 5.1: A Fuzzy Inference System.

\[ R^l : \text{If } x_1 \text{ is } F_{ni} \text{ and } \ldots, x_p \text{ is } F_{pl} \text{ then} \]
\[ y_i = b_{n1}x_i + \ldots + b_{pl}x_p + b_{(p+1)} \text{ for } l = 1, \ldots, M \]

Where \( M \) is the number of rules, \( y_i \) is the output of the \( l \text{th} \) rule, and \( F_{ni}, \ldots, F_{pl} \) are the antecedent fuzzy sets. The overall output of the model is computed by

\[
y = \frac{\sum_{i=1}^{M} w_i y_i}{\sum_{i=1}^{M} w_i} = \frac{\sum_{i=1}^{M} w_i C_i}{\sum_{i=1}^{M} w_i}
\]

Where \( w_i \) is the degree of activation of the antecedent in the \( l \text{th} \) rule

\[
w_i = \prod_{i=1}^{p} F_{ni}(x_i), \text{ } l = 1, 2, \ldots, M
\]

An example of a zeroth-order fuzzy model is shown in figure 5.2 with two input features and two rules.
5.2.1 Fine Tuning of the Rules

After the initial rule base is set, fine tuning of the parameters is needed. The initial model is not optimal because the MFs of the antecedents in the rules are established from the partition of input data only; accordingly, the model cannot appropriately represent the input-output relationship. Adaptive-Network-Based Fuzzy Inference System (ANFIS) method is adopted in the fine tuning of the parameters [Jang J R, 1993]. ANFIS is a neuro-fuzzy approach; i.e., the combination of fuzzy system with neural networks. MATLAB by Math Works has ANFIS in its standard learning method of FIS parameters, and is used in this work [J.M. Mendel, 1995]. Like the initialization of FIS, hybrid learning method is used. First, the parameters of MFs in the antecedent are optimized by gradient descent method (back propagation algorithm). Once the antecedent parameters are fixed, the consequent parameters are estimated by least square method as in the case of the initialization. This alternate step is repeated every epoch; one epoch represents a run of the whole training data set in learning the parameters.

Figure 5.3 shows the convergence curve of classification error and objective function. The objective function is mean-squared error between the desired output and the system output.
Figure 5.3: The convergence of FIS in learning (shown for male data) (a) with four rules and (b) 2 rules. Both classification error (solid line) and the ANFIS objective function (dashed line) are given.

5.3 Trading model optimization

Genetic algorithms are stochastic optimization techniques invented by John Holland [Holland, 1975]. They use ideas taken from biology to guide the search to an optimal, or near optimal, solution. The general approach is to maintain an artificial ecosystem, consisting of a population of chromosomes. Each chromosome represents a possible solution to the general problem. In this case, they take the form of a representation of a particular subset of shares from the index. Attached to each chromosome is a fitness value, which defines how good a solution that chromosome represents. By using mutation, crossover (breeding), and natural selection, the population will converge to one containing only chromosomes with a good fitness. The fitness function in this case is the subset’s tracking error, and the object of the search is to find chromosomes with high fitness values. Many issues arise during the construction of a particular genetic algorithm, all of which affect its performance. Crossover operators,
which take two parent chromosomes and combine them in such a way as to produce a child, need to be carefully designed, to allow the transmission of the best properties of the parents to the child. Mutation is necessary to prevent areas of the search space being discarded, but too high a mutation rate will prevent the desired convergence. The method that is used to update the population in each cycle of the algorithm is another factor. Generational update uses portions of the population to generate enough children to replace the population in its entirety, but more fine grained approaches replace just a few members of the population with children.

The most important of these factors is the design of the crossover operator. This defines the exploratory power of the search, since it is the function that gives rise to the vast majority of new chromosomes. Using mutation alone is not usually sufficient, because of the difficulty of setting a mutation rate which allows for a rapid search, while preventing the search from becoming completely random. Crossover also allows two good chromosomes representing good partial solutions to be combined to form a child representing an even better, more complete, solution. For further information, refer to [Goldberg, 1989].

Optimization is one of the major quantitative tools in the decision making of share market investment. A wide variety of problems in the design, construction, operation and analysis of control etc., can be solved by optimization. The well-known approach to the principle of optimization was first scribbled centuries ago on the walls of an ancient Roman bath-house in
connection with the choice between two aspiring emperors of Rome. It read, “De doubus malis, minus est simper aligendum” which means, “of two evils, always choose the lesser”.

To avoid the increase in computational time, we introduce the concept of genetic algorithms (GA). Genetic algorithms are search procedures whose mechanics are based on those of natural genetics. A genetic algorithm is a search and optimization techniques based on the principle of natural genetics and natural selection. It operates on the principle of the “survival of fittest”, where weak individuals die before reproducing whereas stronger ones survive and bear many off-spring and breed children, which often inherit qualities that are in most cases superior to their parents.

A typical GA is given in figure-5.1, and some points associated with the GA are described as follows:

1. There are many methods to select two parents from the old population, and different GA methods can be obtained by using different selection methods. In this algorithm, the proportionate reproduction operator is employed.

2. In GA, the solution is encoded in a string form. In this chapter, the string of option of each scripts and its encoding is used.
3. Evaluation involves the formation of an objection function, conversion of the objective function into a raw fitness value, conversion of the raw fitness value into a scaled fitness value and conversion of the scaled fitness value into an expected frequency of selection.

4. In reproduction, pairs of individuals are chosen from the population to form a mating pool in such a way that those with high fitness values will be chosen more frequently.

5. Cross over is the most important operator in GA, and the children are generated from each pair of parents. There are several crossover operators, such as single-point crossover [Nicholas (1991)], two-point crossover [Nicholas (1992)] and uniform crossover [Nicholas (1992)].

6. Mutation is also an important operator of GA; it switches randomly one or more bits with some small probability.

7. The chromosome is the complete description of an individual. During crossover, the chromosome will split in half donating half of its individual genes to each of two children.
Figure 5.1 gives the pseudo PASCAL form of a typical genetic algorithm.

*Initialize the parameters of the genetic algorithm:*
*Randomly generate the old-population;*
*For generate I := 1 to max_generation*
*Compute the fitness of each individual in the old_population;*
*Copy the highest fitness of individual to the solution_vector;*
*Use proportionate reproduction method to form mating_pool;*
*While the number_of_individual < population_size do*
  *Select two parents from the mating_pool randomly;*
  *Perform the crossover of the parents to produce two offsprings;*
  *Mutate each offspring based on mutation_probability;*
  *Place the offsprings to new_population;*
*Endwhile*
*Replace the old_population by the new_population;*
*Endfor.*

*Print out the solution_vector as the final solution.*

Figure 5.1. Outline of a typical pseudo PASCAL form of the single genetic algorithm.
5.3.1 Genetic Algorithm Operators

5.3.1.1 Reproduction

The commonly used reproduction operator is the proportionate reproduction operator where a string is selected for the mating pool with a probability proportional to its fitness value \( F \). Thus, the \( i \)th string in the population is selected with the difference between the probability proportional to \( G_i \) (the gain value (Return on Investment) attained in the \( i \)th type of scripts) and the probability proportional to \( R_i \) (the risk value attained in the \( i \)th type of scripts). Since, the population size is usually kept fixed in a simple genetic algorithm, the sum of the probability for selecting the \( i \)th string is

\[
F(i) = \frac{G_i - R_i}{\sum_{k=1}^{n} G_k + \sum_{k=1}^{n} R_k}
\]

Where \( n \) is the population size. Here the size of the population is 11, which are the different types of BSE scripts. Each digit of a string varies from 1 to 3 which represented by the high, medium and low investment. Usually, the risk value is evaluated by the way of establishing statistical range value, which is taken from its fluctuation of that particular sub-index. The value of higher range is probably the high risk and the low range will be the low risk.
5.3.1.2 Crossover

In the crossover operator, two strings are selected from the mating pool and a portion of the strings are exchanged between these strings. For example, the crossover operator applied to the following two strings assuming the crossover site is 6

String 1: 5 8 3 10 7 1 6 11 9 4 2
String 2: 7 4 2 5 11 3 10 1 8 6 9

After crossover the above two strings are changed in the following manner:

String 1: 5 8 3 10 7 1 4 2 11 6 9
String 2: 7 6 11 5 9 3 10 1 8 4 2

(Here, 1 to 11 represented by script number and each script number may be represented by 1 or 2 or 3 the levels of investment namely high, medium and low)

5.3.1.3 Mutation

Mutation is also an important GA operator. It switches randomly one or more bits with some small probability. For example, when the string

5 8 3 10 7 1 4 2 11 6 9

is mutated between the sites 4 and 10, we have

5 8 3 6 7 1 4 2 11 10 9
The working principle for one generation is illustrated in table – 5.1.

The selected strings are rearranged in the order given below:

Random numbers for selecting sites to rearrangement of strings are given by

\[
10 \ 4\ 5\ 8\ 7\ 1\ 3\ 2\ 6\ 9
\]

The job sequences in table 5.2 – 5.4 give the input to the next generation of the genetic algorithm. In each generation, we pick the best sequence and record it. The results of the first 20 best sequences in their corresponding generations are given in table -5.3.

In table-5.5, we observe that after the 16\textsuperscript{th} generation, the fitness value increased slowly, and the best sequence 10(H) 8(M) 7(H) 1(H) 11(H) 9(M) 5(M) 4(M) 2(L) 6(L) 3(L) is obtained from the 16\textsuperscript{th} generation. In the same table, we show that the first genetic algorithm is evaluate the first best sequence of investment 5-9-11-1-4-2-7-10-6-3-8 and then it pass on to the next genetic algorithm and this algorithm is able to evaluate the level of investment like, high, medium, and low. After pass on to the first and second generating function, the best sequence and the best level of investment is obtained from 13\textsuperscript{th} iteration and its fitness value is 10245. Similarly, the second, third iterations are evaluated and only the final result is put on the table-5.5.
### Table – 5.3, After Crossover and before Mutation

<table>
<thead>
<tr>
<th>String Number</th>
<th>Script Sequence</th>
<th>Random Number for Mutation</th>
<th>Yes / No For Mutation</th>
<th>Random Number for Mutation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>11 5 9 7 10 4 8 2 6 3 1</td>
<td>0.64</td>
<td>N</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>7 5 1 1 9 2 4 1 0 1 6 8 3</td>
<td>0.87</td>
<td>N</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td>7 5 8 9 4 2 1 0 1 6 1 1 3</td>
<td>0.02</td>
<td>Y</td>
<td>4,8</td>
</tr>
<tr>
<td>4</td>
<td>8 9 7 5 1 1 2 4 6 1 0 3 1</td>
<td>0.06</td>
<td>Y</td>
<td>1,7</td>
</tr>
<tr>
<td>5</td>
<td>9 5 4 1 0 2 3 6 7 1 1 8</td>
<td>0.41</td>
<td>N</td>
<td>-</td>
</tr>
<tr>
<td>6</td>
<td>9 7 5 1 1 1 4 2 1 0 6 3 8</td>
<td>0.65</td>
<td>N</td>
<td>-</td>
</tr>
<tr>
<td>7</td>
<td>5 9 4 2 7 1 0 1 6 1 1 8 3</td>
<td>0.84</td>
<td>N</td>
<td>-</td>
</tr>
<tr>
<td>8</td>
<td>5 9 4 2 7 1 0 1 6 1 1 8 3</td>
<td>0.32</td>
<td>N</td>
<td>-</td>
</tr>
<tr>
<td>9</td>
<td>9 7 2 5 4 1 0 1 6 3 1 1 8</td>
<td>0.66</td>
<td>N</td>
<td>-</td>
</tr>
<tr>
<td>10</td>
<td>9 2 7 5 1 1 4 1 0 6 3 8</td>
<td>0.60</td>
<td>N</td>
<td>-</td>
</tr>
</tbody>
</table>
The same procedure can also be applied to find the best sequence of investment level, like high, medium, and low. This will take care of the next genetic algorithm. The next table elaborates the procedure to evaluate the optimum sequence of the investing level, it could be specified with in brackets.

<table>
<thead>
<tr>
<th>String Number</th>
<th>Script Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>11 5 9 7 10 4 8 2 6 3 1</td>
</tr>
<tr>
<td>2</td>
<td>7 5 1 1 9 2 4 1 0 1 6 8 3</td>
</tr>
<tr>
<td>3</td>
<td>7 5 8 1 4 2 1 0 9 6 1 1 3</td>
</tr>
<tr>
<td>4</td>
<td>4 9 7 5 1 1 2 7 6 1 0 3 1</td>
</tr>
<tr>
<td>5</td>
<td>9 5 4 1 0 2 3 6 7 1 1 1 8</td>
</tr>
<tr>
<td>6</td>
<td>9 7 5 1 1 4 2 1 0 6 3 8</td>
</tr>
<tr>
<td>7</td>
<td>5 9 4 2 7 1 0 1 6 1 1 8 3</td>
</tr>
<tr>
<td>8</td>
<td>5 9 4 2 7 1 0 1 6 1 1 8 3</td>
</tr>
<tr>
<td>9</td>
<td>9 7 2 5 4 1 0 1 6 3 1 1 8</td>
</tr>
<tr>
<td>10</td>
<td>9 2 7 5 1 1 4 1 0 6 3 8</td>
</tr>
</tbody>
</table>
Table 5.5: Best sequence of scripts, Fitness Value and its generations

<table>
<thead>
<tr>
<th>Generation</th>
<th>Best sequence of scripts</th>
<th>Fitness Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5(M) 9(M) 11(M) 1(L) 4(H) 2(L) 7(M) 10(M) 6(L) 3(L) 8(L)</td>
<td>7412</td>
</tr>
<tr>
<td>2</td>
<td>5(M) 9(M) 11(L) 1(L) 4(M) 2(L) 7(H) 10(H) 6(L) 3(H) 8(M)</td>
<td>7548</td>
</tr>
<tr>
<td>3</td>
<td>5(M) 9(H) 11(M) 1(L) 4(M) 2(H) 7(M) 10(M) 6(L) 3(H) 8(L)</td>
<td>7624</td>
</tr>
<tr>
<td>4</td>
<td>5(H) 9(M) 11(M) 1(H) 4(H) 2(L) 7(L) 10(M) 6(H) 3(M) 8(L)</td>
<td>7666</td>
</tr>
<tr>
<td>5</td>
<td>5(H) 9(L) 11(M) 1(L) 4(H) 2(L) 7(M) 10(H) 6(L) 3(H) 8(L)</td>
<td>7754</td>
</tr>
<tr>
<td>6</td>
<td>5(H) 9(M) 11(M) 1(L) 4(H) 2(L) 7(M) 10(H) 6(L) 3(H) 8(L)</td>
<td>7898</td>
</tr>
<tr>
<td>7</td>
<td>5(M) 9(L) 11(M) 1(L) 4(M) 2(M) 7(L) 10(L) 6(H) 3(H) 8(M)</td>
<td>7899</td>
</tr>
<tr>
<td>8</td>
<td>5(H) 9(M) 11(H) 1(L) 4(H) 2(L) 7(M) 10(H) 6(L) 3(L) 8(L)</td>
<td>9321</td>
</tr>
<tr>
<td>9</td>
<td>5(L) 9(H) 11(H) 1(L) 4(H) 2(L) 7(M) 10(H) 6(M) 3(H) 8(L)</td>
<td>9411</td>
</tr>
<tr>
<td>10</td>
<td>5(L) 9(H) 11(H) 1(L) 4(H) 2(M) 7(L) 10(L) 6(L) 3(H) 8(H)</td>
<td>9564</td>
</tr>
<tr>
<td>11</td>
<td>5(H) 9(M) 11(H) 1(L) 4(H) 2(L) 7(M) 10(L) 6(L) 3(L) 8(H)</td>
<td>9777</td>
</tr>
<tr>
<td>12</td>
<td>5(L) 9(H) 11(M) 1(M) 4(H) 2(H) 7(H) 10(M) 6(L) 3(H) 8(H)</td>
<td>9898</td>
</tr>
<tr>
<td>13</td>
<td>5(H) 9(M) 11(M) 1(M) 4(H) 2(L) 7(H) 10(M) 6(M) 3(H) 8(L) 10245*</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>5(L) 9(M) 11(M) 1(L) 4(H) 2(L) 7(M) 10(M) 6(M) 3(H) 8(L)</td>
<td>8565</td>
</tr>
<tr>
<td>15</td>
<td>5(M) 9(M) 11(M) 1(L) 4(H) 2(H) 7(L) 10(M) 6(M) 3(L) 8(M)</td>
<td>8179</td>
</tr>
<tr>
<td>16</td>
<td>5(H) 9(M) 11(M) 1(L) 4(H) 2(H) 7(M) 10(H) 6(M) 3(H) 8(M)</td>
<td>7958</td>
</tr>
<tr>
<td>17</td>
<td>5(H) 9(M) 11(L) 1(H) 4(H) 2(L) 7(M) 10(H) 6(L) 3(H) 8(L)</td>
<td>7722</td>
</tr>
<tr>
<td>18</td>
<td>5(H) 9(H) 11(L) 1(H) 4(H) 2(H) 7(M) 10(H) 6(H) 3(M) 8(L)</td>
<td>7523</td>
</tr>
<tr>
<td>19</td>
<td>5(M) 9(M) 11(M) 1(L) 4(H) 2(L) 7(H) 10(M) 6(L) 3(H) 8(L)</td>
<td>7189</td>
</tr>
<tr>
<td>20</td>
<td>5(M) 9(L) 11(M) 1(M) 4(H) 2(L) 7(M) 10(H) 6(L) 3(M) 8(M)</td>
<td>6700</td>
</tr>
<tr>
<td>2</td>
<td>9(M) 5(M) 10(M) 4(H) 1(M) 2(M) 6(M) 11(M) 7(M) 3(M) 8(M)</td>
<td>10578</td>
</tr>
<tr>
<td>3</td>
<td>5(H) 9(H) 10(M) 4(H) 1(M) 2(L) 11(L) 7(M) 3(M) 6(M) 8(M)</td>
<td>10910</td>
</tr>
<tr>
<td>4</td>
<td>5(H) 9(H) 11(M) 4(H) 1(M) 2(M) 10(M) 7(M) 3(M) 6(M) 8(M)</td>
<td>11547</td>
</tr>
</tbody>
</table>
In the above table, the first genetic algorithm is to find the best sequence of the first iteration and then it pass on to the second genetic algorithm. The second algorithm is going to evaluate the amount of investment in each sub-indexes and the fitness function will give the corresponding return on investment. Initially, the fitness value is going on increasing up to 13th iteration and then the value is reduced after that. Therefore, these two genetic algorithms are acted as the nested loop structure. After completing the first iteration of upper genetic algorithm, it should find the best sequence for that iteration and simultaneously it can be evaluated that what should be its investment level of each sub-index.
5.4 Conclusion

Studies have shown that, regardless of the investor's base market and behavior, types of portfolios selection, over longer / shorter periods, offer higher returns at lower risk investments in national markets only.

The program part of the genetic algorithm is set to a maximum of 500 generations in genetic algorithm-1 and 300 from genetic algorithm-2, from which, we observe that the return on investment goes on increasing from generation -1 to generation -17 and then decreasing from 17 to the rest of the generations. Hence, the optimum sequence and the optimum distribution of amount is given by 10(H) 8(M) 7(H) 1(H) 11(H) 9(M) 5(M) 4(M) 2(L) 6(L) 3(L) and the maximum return on investment is Rs.17,258/-. This is the amount we will get after one year if we invested the amount is Rs.12,000/-.

In this chapter, two genetic algorithms are used to select the optimum sequence of investment strategy and the level of investment. If the investors want to invest their amount in one or two scripts then the order will give the best way of investing their money with less risk. The optimum sequence gives increasing order of return on investment with increasing order of risk factor.