*International Journal of Advanced Scientific Technologies ,Engineering and Management Sciences (IJASTEMS-ISSN: 2454-356X) Volume.3,Special Issue.1,March.2017*

# Metaheuristic Algorithms for Feature Selection in Data Classification

***S.Vani Kumari*, Dr.K.Usha Rani**,***
**** Dept of CSE, GMRIT,Rajam&Reasearch Scholar @ Sri PadmavathyMahilaUniversity,Tirupathi, AP,India***
***** Department of Computer Science ,Sri PadmavathyMahila University , Tirupathi,AP, India***

***ABSTRACT***

***Feature selection is process of selecting a subset of extracted features that are most relevant for the model construction in data classification. Feature selection is adapted for improving the accuracy of classification, simplifying the classification process and hence reduce the time involved in training the samples. Data generally contains some features that are irrelevant or redundant and hence can be discarded without any loss of essential information. Feature selection aims at reducing the high-dimensional feature space to a low-dimensional feature space resulting in an optimal subset of available features. Feature selection methods are broadly divided into three categories namely wrappers, filters and embedded methods..A metaheuristic is a high level procedure that is designed to generate or find solution for an optimization problem and is used as a strategy to guide the searching process. The aim of a metaheuristic algorithm is to explore the entire search space to find the near optimal solutions. The aim of this paper is to study various metaheuristic algorithms for the feature selection namely Tabu Search, Genetic Algorithms, Scatter search.***

***Keywords:FeatureSelection,TabuSearch,GeneticAlgorithm,ScatterSearch.***

## I.INTRODUCTION

Data in the modern world is becoming more and more high dimensional.Classification is a data mining task that assigns items in a collection to target groups or classes. The goal of classification is to correctly predict the target class for each case in the data. For example, a classification model could be used to identify a patient is at low, medium, or high risk of a particular disease.A classification task begins with a data set in which the class labels arealready known. For example,a classification model that predicts credit risk might be developed based on observed data for many loan applicants over a period of time. In addition to the historical credit rating, the data might track employment history, home ownership or rental, years of residence, number and type of investments, and so on. Credit rating would be the target, the other attributes would be the predictors, and the data for each customer would constitute a case.

The simplest type of classification problem is binary classification. In binary classification, the target attribute has only two possible values. In the model build (training) process, a classification algorithm finds relationships between the values of the predictors and the values of the target. Different classification algorithms use different techniques for finding relationships. These relationships are summarized in a model, which can then be applied to a different data set in which the class assignments are unknown Classification models are tested by comparing the predicted values to known target values in a set of test data. The historical data for a classification project is typically divided into two data sets: one for building the model; the other for testing the model. A classification model is tested by applying it to test data with known target values and comparing the predicted values with the known values.

The test data must be compatible with the data used to build the model and must be prepared in the same way that the build data was prepared. Typically the build data and test data come from the same historical data set. A percentage of the records is used to build the model; the remaining records are used to test the model. Test metrics are used to assess how accurately the model predicts the known values. If the model performs well and meets the business requirements, it can then be applied to new data to predict the future. Accuracy refers to the percentage of correct predictions made by the model when compared with the actual classifications in the test data. ROC is another metric for comparing predicted and actual target values in a classification model. ROC, like lift, applies to binary classification and requires the designation of a positive class.

## II.FEATURE SELECTION

Feature selection is the process of decreasing the inputs for processing and analysis, or of finding the most relevant input i.e. it refers to the process of extracting useful information or features from existing data.Feature selectionto some extent is degree of cardinality reduction, to impose a cut-off on the number of attributes that can be considered when building a model. Data contains more information than that is needed to build the model, or sometimes unnecessary information. For example, in a dataset with 600 columns that describe the characteristics of customers; however, if the data in some of the columns is very sparse that we would gain little benefit from adding them to the model, and if some of the columns duplicate each other, using both columns could affect the model.

Feature selection aims at :

(i)reduction in the size of problem i.e. reduce computing time and space required to run algorithms.

(ii)improving classifiers. Initially by removing noisy or irrelevant features. Finally by reducing the likelihood of overfitting to noisy data.

(iii)identifying features which be relevant to a specific problem.

Feature selection algorithms fall under two categories: feature ranking and set selection[GA-2]. Feature ranking eliminates all features that do not achieve an adequate score and rank the features by metric. Set selection searches for the optimal set from the set of possible

features. Feature selection (known as set selection) is a method used in machine learning, wherein for application of learning algorithm subsets of the available features are selected from data. The most effective set contains the smallest range of dimensions that contributes to accuracy; one discards the remaining unimportant dimensions. This is a stage of preprocessing which is very important and is one of two ways by which curse of dimensionality is avoided (the other is feature extraction) . These are two approaches: Forward selection: Begin with no variables and add them one by one, at each step adding the one that decreases the error, until any more addition does not considerably decrease the error. Backward selection: Begin with all the variables and eliminate them one by one, at each step removing the one that decreases the error (or will increase it slightly); until any more removal increase the error considerably.

In the context of classification, feature selection techniques can be organized into three categories, depending on how they combine the feature selection search with the construction of the classification model: filter methods, wrapper methods and embedded methods.

Filter method selects variables regardless of the model. In this filter method a best subset of the features are selected and then tested on the learning algorithm. They are based only on general features like the correlation with the variable to predict. Filter methods suppress the least interesting variables [1]The other variables will be part of a classification or a regression model used to classify or to predict data. These methods are particularly effective in computation time and robust to overfitting. However, filter methods tend to select redundant variables because they do not consider the relationships between variables. Therefore, they are mainly used as a pre-process method.Filter techniques assess the relevance of features by looking only at the intrinsic properties of the data. In most cases a feature relevance score is calculated, and low-scoring features are removed. Afterwards, this subset of features is presented as input to the classification algorithm. Advantages of filter techniques are that they easily scale to very high-dimensional datasets, they are computationally simple and fast, and they are independent of the classification algorithm. As a result, feature selection needs to be performed only once, and then different classifiers can be evaluated. Examples of filter methods include feature selection based on Euclidian distance, chi-square, gain ratio [2], correlation based, Markov blanket filter[3], Fast correlation-based feature selection [4].

Wrapper methods evaluates subsets of variables to detect the interactions between variables. The drawback of using wrapper methods is increase in over fitting if the number of samples in less and consumes considerable computation when there are large number of variables. Wrappers can be computationally expensive because model training and cross-validation must be repeated over each feature subset, and the outcome is tailored to a particular model. Some of the wrapper methods include Sequential forward selection, F-Score,sequential backward selection[5],simulated annealing, genetic algorithms .

An embedded method combines the advantage of both the wrapper and filter methods and does its own variable selection process and performs feature selection and classification simultaneously. This method models the feature dependencies. Decision trees, weighted naive Bayes and Feature selection using
the weight vector of SVM are some examples of embedded methods.

III.TABU SEARCH

The tabu search, proposed by Glover[6], is a meta heuristic method that can be used to solve combinatorial optimization problem. It has received widespread attention recently. Its flexible control framework and several spectacular successes in solving NP-hard problems caused rapid growth in its application. It differs from the local search technique in the sense that tabu search allows moving to a new solution which makes the objective function worse in the hope that it will not trap in local optimal solutions. Tabu search uses a short-term memory, called tabu list, to record and guide the process of the search. In addition to the tabu list, we can also use a long-term memories and other prior information about the solutions to improve the intensification and/or diversification of the search..Local searches take a potential solution to a problem and check its immediate neighbors in the hope of finding an improved solution. Local search methods have a tendency to become stuck in suboptimal regions or on plateaus where many solutions are equally fit.Tabu search enhances the performance of local search by relaxing its basic rule. First, at each step worsening moves can be accepted if no improving move is available. In addition, prohibitions are introduced to discourage the search from coming back to previously-visited solutions.The implementation of tabu search uses memory structures that describe the visited solutions or user-provided sets of rules. If a potential solution has been previously visited within a certain short-term period or if it has violated a rule, it is marked as "tabu" (forbidden) so that the algorithm does not consider that possibility repeatedly.

The tabu search scheme can be outlined as follows: start with an initial (current) solution x, called a configuration, evaluate the criterion function for that solution. Then, follow a certain set of candidate moves, called the neighborhood N(x) of the current solution x. If the best of these moves is not tabu (i.e., not in the tabu list) or if the best is tabu, but satisfies the aspiration criterion, which will be explained below, then pick that move and consider it to be the new current solution. Repeat the procedure for a certain number of iterations. On termination, the best solution obtained so far is the solution of the tabu search. Note that the solution that is picked at certain iteration is put in the tabu list so that it is not allowed to be reversed in the next l iterations, i.e., this solution is tabu. The l is the size of . When the length of tabu list reaches that size, then the first solution on the ¹¸ is freed from being tabu and the new solution enters that list. The process continues. The ¹¸ acts as a shortterm

memory. By recording the history of searches, tabu search can control the direction of the following searches. The aspiration criterion could reflect the value of the criterion function, i.e., if the tabu solution results in a value of the criterion function that is better than the best known so far, then the aspiration criterion is satis- Tabu search is a metaheuristic algorithm that can be used for solving combinatorial optimization problems.

Representation of Solutions. A solution s is a subset of features. It is represented by a bit string of size n, the total number of features: s = [a1, ..., an] with $a_i \in \{0, 1\}$, $\forall i \in \{1, ..., n\}$. The ithbit aiindicates if the feature iis chosen($a_i = 1$) or, on the contrary, if it is not ($a_i = 0$).

Evaluation of Solutions. For the FS problem in classification, several criteria are commonly used to measure the quality of a solution. First, it may be measured by the quality of the classification realized using the selected features. Most of classifiers propose to compute the accuracy, which is defined as the ratio between the well-classified observations and the total number of observations tested. The accuracy is computed as

accuracy= number of well-classified observations/total number of observations

Secondly, the number of selected features is an important criterion for FS problem. Indeed, in order to obtain more interpretable models, the number of selected features should be minimized. This criterion is defined as the ratio between the number of selected features (# S Features) and the total number of features (# Features). In order to obtain a maximization criterion, the criterion, noted features, is defined as follows:

features= 1− (# S Features / # Features)

In local search algorithms and in particular in Tabu Search, the exploration of the neighborhood of a solution can be time-consuming. Indeed, in the original Tabu Search method, all the non-tabu neighbors of a solution are evaluated at each iteration. In the FS problem, the evaluation of a solution is computed by applying a classification procedure like KNN,SVM etc. This one can be computationally expensive when the number of observations and/or features becomes large. Hence, the evaluation of the whole neighborhood at each iteration can not be considered. Following is the pseudo code for the Tabu search:

```
sBest ← s0
tabuList ← [ ]
while (not stoppingCondition())
        candidateList ← []
        bestCandidate ← null
        for (sCandidate in sNeighborhood)
                if ( (not tabuList.contains(sCandidate))
and (fitness(sCandidate) > fitness(bestCandidate)) )
                        bestCandidate ← sCandidate
                end
        end
if (fitness(bestCandidate) > fitness(sBest))
                sBest ← bestCandidate
        end
        tabuList.push(bestCandidate);
        if (tabuList.size > maxTabuSize)
                tabuList.removeFirst()
        end
end
return sBest
```

Tabu search uses a local or neighborhood search procedure to iteratively move from one potential solution x to an improved solution x' in the neighborhood of x , until some stopping criterion has been satisfied (generally, an attempt limit or a score threshold). Local search procedures often become stuck in poor-scoring areas or areas where scores plateau. In order to avoid these pitfalls and explore regions of the search space that would be left unexplored by other local search procedures, tabu search carefully explores the neighborhood of each solution as the search progresses. The solutions admitted to the new neighborhood $N^*(x)$, are determined through the use of memory structures. Using these memory structures, the search progresses by iteratively moving from the current solution x to an improved solution x' in $N^*(x)$, .

These memory structures form what is known as the tabu list, a set of rules and banned solutions used to filter which solutions will be admitted to the neighborhood to be explored by the search. In its simplest form, a tabu list is a short-term set of the solutions that have been visited in the recent past (less than iterations ago, where is the number of previous solutions to be stored - is also called the tabu tenure). More commonly, a tabu list consists of solutions that have changed by the process of moving from one solution to another. It is convenient, for ease of description, to understand a "solution" to be coded and represented by such attributes.

The memory structures used in tabu search can roughly be divided into three categories

- Short-term: The list of solutions recently considered. If a potential solution appears on the tabu list, it cannot be revisited until it reaches an expiration point.
- Intermediate-term: Intensification rules intended to bias the search towards promising areas of the search space.
- Long-term: Diversification rules that drive the search into new regions (i.e. regarding resets when the search becomes stuck in a plateau or a suboptimal dead-end).

## IV.SCATTER SEARCH

Scatter search (SS) was first introduced in Glover (1977)[8] as a heuristic for integer programming. Scatter search is an evolutionary method that has been successfully applied to hard optimization problems. The fundamental concepts and principles of the method were first proposed in the 1970s, based on formulations dating back to the 1960s for combining decision rules and problem constraints. In contrast to other evolutionary methods like genetic algorithms, scatter search is founded on the premise that systematic designs and methods for creating new solutions afford significant benefits beyond

those derived from recourse to randomization. It uses strategies for search diversification and intensification that have proved effective in a variety of optimization problems. Scatter search is a Metaheuristic and a Global Optimization algorithm. It is also sometimes associated with the field of Evolutionary Computation given the use of a population and recombination in the structure of the technique. Scatter Search is a sibling of Tabu Search, developed by the same author and based on similar origins. The objective of Scatter Search is to maintain a set of diverse and high-quality candidate solutions. The principle of the approach is that useful information about the global optima is stored in a diverse and elite set of solutions (the reference set) and that recombining samples from the set can exploit this information. The strategy involves an iterative process, where a population of diverse and high-quality candidate solutions that are partitioned into subsets and linearly recombined to create weighted centroids of sample-based neighbourhoods. The results of recombination are refined using an embedded heuristic and assessed in the context of the reference set as to whether or not they are retained.

Scatter search incorporates an implicit form of memory as a result of the interactions among the Reference Set Update, the Solution Combination Method and the Subset Generation Method. The Reference Set Update, in its most basic form, is designed to "remember" the best solutions encountered during the search. Selected features of these solutions provide the basis for creating new trial solutions with the Combination Method. Hence, the overall process is instrumental in the transmission of information embedded in the reference solutions.

The scatter search methodology is very flexible, since each of its elements can be implemented in a variety of ways and degrees of sophistication[9]. A basic design to implement scatter search based on the well-known ''five methods''. The advanced features of scatter search are related to the way these five methods are implemented. That is, the sophistication comes from the implementation of the SS methods instead of the decision to include or exclude some elements (like in the case of tabu search, as mentioned above). The fact that the mechanisms within scatter search are not restricted to a single uniform design allows the exploration of strategic possibilities that may prove effective in a particular implementation. These observations and principles lead to the following template for implementing scatter search that consists of five methods.

1. A diversification generation method to generate a collection of diverse trial solutions, using an arbitrary trial solution (or seed solution) as an input.
2. An improvement method to transform a trial solution into one or more enhanced trial solutions. (Neither the input nor the output solutions are required to be feasible, though the output solutions will more usually be expected to be so. If no improvement of the input trial solution results, the ''enhanced'' solution is considered to be the same as the input solution.)
3. A reference set update method to build and maintain a reference set consisting of the b ''best'' solutions found (where the value of b is typically small, e.g., no more than 20), organized to provide efficient accessing by other parts of the method. Solutions gain membership to the reference set according to their quality or their diversity.
4. A subset generation method to operate on the reference set, to produce a subset of its solutions as a basis for creating combined solutions.
5. A solution combination method to transform a given subset of solutions produced by the subset generation method into one or more combined solution vectors.

## V.GENETIC ALGORITHM

In feature selection, genetic algorithm[11] is used as a random selection algorithm, capable of effectively exploring large search spaces, which is usually required in case of attribute selection. If the original feature set contains an N number of features, the total number of competing candidate subsets to be generated is 2N, which is a hugenumber even for medium sized N. further unlike many search algorithms, which perform a local, greedy search, GA perform a global search. Genetic algorithms have demonstrated substantial improvement over a variety of random and local search methods .this is accomplished by their ability to exploit accumulating information about an initially unknown search space in order to bias subsequent search in to promising subspaces.

A genetic algorithm (GA) is a metaheuristic inspired by the process of natural selection that belongs to the larger class of evolutionary algorithms (EA). Genetic algorithms are commonly used to generate high-quality solutions to optimization and search problems by relying on bio-inspired operators such as mutation, crossover and selection. In a genetic algorithm, a population of candidate solutions (called individuals, creatures, or phenotypes) to an optimization problem is evolved toward better solutions. Each candidate solution has a set of properties (its chromosomes or genotype) which can be mutated and altered; traditionally, solutions are represented in binary as strings of 0s and 1s, but other encodings are also possible.

The evolution usually starts from a population of randomly generated individuals, and is an iterative process, with the population in each iteration called a generation. In each generation, the fitness of every individual in the population is evaluated; the fitness is usually the value of the objective function in the optimization problem being solved. The more fit individuals are stochastically selected from the current population, and each individual's genome is modified (recombined and possibly randomly mutated) to form a new generation. The new generation of candidate solutions is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population[12].

1.[Start] Generate random population of n chromosomes (suitable solutions for the problem)

2.[Fitness] Evaluate the fitness f(x) of each chromosome x in the population

3.[New population] Create a new population by repeating following steps until the new population is complete

(i)[Selection] Select two parent chromosomes from a population according to their fitness (the better fitness, the bigger chance to be selected)

(ii)[Crossover] With a crossover probability cross over the parents to form new offspring (children). If no crossover was performed, offspring is the exact copy of parents.

(iii)[Mutation] With a mutation probability mutate new offspring at each locus (position in chromosome).

(iv)[Accepting] Place new offspring in the new population

4.[Replace] Use new generated population for a further run of the algorithm

5.[Test] If the end condition is satisfied, stop, and return the best solution in current population

6.[Loop] Go to step 2

## VI.CONCLUSION

Feature selection in important in any data mining task.Feature selection methods aims to identify and remove unnecessary ,irrelevant and redundant attributes from data which do not contribute to the accuracy of a predictive model or may in fact adversely effect on the accuracy of the model being used.Fewer attributes is desirable as it reduces the complexity of the model and a simpler model is in deedsimpler to understand and can be applied for various applications. This paper discusses three meta heuristic algorithms for feature selection. All the algorithms presented in the paper are population based methods and are used in many diversified applications

## REFERENCES

[1]Saeys, Yvan, IñakiInza, and Pedro Larrañaga. "A review of feature selection techniques in bioinformatics." *bioinformatics* 23.19 (2007): 2507-2517.

[2] Karegowda, Asha Gowda, A. S. Manjunath, and M. A. Jayaram. "Comparative study of attribute selection using gain ratio and correlation based feature selection." *International Journal of Information Technology and Knowledge Management* 2.2 (2010): 271-277.

[3] Fu, Shunkai, and Michel C. Desmarais. "Markov blanket based feature selection: a review of past decade." *Proceedings of the world congress on engineering*. Vol. 1. 2010.

[4] Yu, Lei, and Huan Liu. "Feature selection for high-dimensional data: A fast correlation-based filter solution." *ICML*. Vol. 3. 2003.

[5] Huang, Yi-Jhe, et al. "Automated feature set selection and its application to MCC identification in digital mammograms for breast cancer detection." Sensors 13.4 (2013): 4855-4875.

[6] Zhang, Hongbin, and Guangyu Sun. "Feature selection using tabu search method." Pattern recognition 35.3 (2002): 701-711.

[7] Mousin, Lucien, et al. "Feature Selection using Tabu Search with Learning Memory: Learning Tabu Search." International Conference on Learning and Intelligent Optimization. Springer International Publishing, 2016

[8] Larson, Jeffrey, and Francis Newman. "An implementation of scatter search to train neural networks for brain lesion recognition." Involve, a Journal of Mathematics 4.3 (2012): 203-211.

[9] Martí, Rafael, Manuel Laguna, and Fred Glover. "Principles of scatter search." European Journal of Operational Research 169.2 (2006): 359-372.

[10] Nepomuceno, Juan A., Alicia Troncoso, and Jesús S. Aguilar-Ruiz. "Scatter search-based identification of local patterns with positive and negative correlations in gene expression data." Applied Soft Computing 35 (2015): 637-651.

[11] Amarnath, B., and S. Appavu alias Balamurugan. "Metaheuristic Approach for Efficient Feature Selection: A Data Classification Perspective." Indian Journal of Science and Technology 9.4 (2016).

[12] Mansoori, Tanzeem Khan, AmritSuman, and Sadhana K. Mishra. "Feature Selection by Genetic Algorithm and SVM Classification for Cancer Detection." International Journal 4.9 (2014).