CLASSIFICATION OF DERMATOLOGY USING ANT COLONY OPTIMIZATION (ACO) WITH FEATURE SELECTION AND GENETIC ALGORITHM (GA)

AYE THIRI MON

M.C.Sc. (THESIS)

AUGUST, 2016

2.5.5 Ant Colony System

First, it exploits the search experience accumulated by the ants more strongly than AS does through the use of a more aggressive action choice rule. Second, pheromone evaporation and pheromone deposit take place only on the arcs belonging to the best-so-far tour. Third, each time an ant uses an arc (i,j) move from city i to city j, it removes some pheromone from the arc to increase the exploration of alternative paths.

Artificial ants iteratively sample tours through a loop that includes a tour construction biased by the artificial pheromone trails and the heuristic information. The main mechanism at work in ACO algorithms that triggers the discovery of good tours is the positive feedback given through the pheromone update by the ants: the shorter the ant's tour, the higher the amount of pheromone the ant deposits on the arcs of its tour.

This in turn leads to the fact that these arcs have a higher probability of being selected in the subsequent iterations of the algorithm. The emergence of arcs with high pheromone values is further reinforced by the pheromone trail evaporation that avoids an unlimited accumulation of pheromones and quickly decreases the pheromone level on arcs that only very rarely, or never, receives additional pheromone.

2.6 Genetic Algorithm

Genetic algorithms attempt to incorporate ideas of natural evolution. In general, genetic learning starts as follows. An initial population is created consisting of randomly generated rules. Each rule can be represented by a string of bits. As a simple example, suppose that samples in a given training set are described by two Boolean attributes, A1 and A2, and that there are two classes, C1 and C2. The rule "IF A1 AND NOT A2 THEN C2 " can be encoded as the bit string "100," where

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude and sincere thanks to all persons who contributed directly or indirectly towards the success of this thesis.

I owe my respectful thanks to **Dr. Win Aye**, the Rector, Computer University (Mandalay), for her kind permission to make this thesis in this university.

I am deeply thankful to **Dr. Thi Thi Soe**, Associate Professor, Head of Software Department, Computer University (Mandalay), for her continuous, sincere, and kind guidance during the period of completing this thesis.

I am greatly indebted to my supervisor, **Dr Thandar Aung**, Associate Professor, Head of Information Science Department, Computer University (Mandalay), for her valuable supervision, good advice, detailed guidance, and helpful suggestions throughout the terms of finishing the thesis.

I would like to extend my deepest gratitude to **U Thaung Kyaw**, Associate Professor, Head of Department of English at Computer University (Mandalay), for editing my thesis.

Finally, I also thank all the members of Computer University (Mandalay), who attended the seminars on this thesis, for their support, valuable suggestions, helpful hints, and fair criticisms.

Moreover, I also thank all my **mentors and friends**, for their invaluable suggestions and helpful contributions to success of this thesis;

Especially, my deepest thanks go to **my parents** for giving me moral and material support, sympathy and empathy, care and kindness throughout my studies and my efforts to complete this thesis without any trouble.

ABSTRACT

Feature selection is used to find the good subset of feature to improve the classifier's accuracy. In this thesis, firstly, preprocessing step is used to fill the missing value by using mean value and normalize the data value by Min_Max normalization. Secondly, Ant Colony Optimization (ACO) is used for feature selection with filter approach. Thirdly, Genetic Algorithm (GA) is used for classification. Finally, performance of classifier is calculated by using cross validation method. Dermatology Dataset is used to implement the system. The data set is mainly used from University of California at Irvine (UCI) repository of machine learning databases. This thesis is implemented by C#.Net programming.

CONTENTS

LIST OF FIGURES

LIST OF EQUATIONS

LIST OF TABLES

CHAPTER 1
INTRODUCTION

Nowadays, technology provides a great boost to use computerized systems for industry, medicines, education, and even small local businesses and so on.

Database technology has evolved from primitive file processing to the development of database management systems with query and transaction processing. Further progress has led to the increasing demand for efficient and effective advanced data analysis tools. This need is a result of the explosive growth in data collected from applications, including business and management, government administration, science and engineering and environmental control.

Data mining is the task of discovering interesting patterns from large amounts of data, where data can be stored in databases, data warehouses, or other information repositories. It is a young interdisciplinary field, drawing from areas such as database systems, data warehousing, statistic, machine learning, data visualization, information retrieval, and high performance computing. Other contributing areas include neural networks, pattern recognition, spatial data analysis, image databases, signal processing, and many application fields, such as business, economics, and bioinformatics.

Ant Colony Optimization (ACO) is used for selection of optimal feature subsets for dermatology data set. Ant Colony Optimization (ACO) is used to solve discrete combinatorial optimization problems. The system used filter approach of feature selection method to find the good subset of feature to improve the classifier's accuracy.

Genetic algorithm attempts to incorporate ideas of natural evolution. Genetic algorithm has been used for classification as well as other optimization problems.

1.1 Background History of the Application

The practice of dermatology has, over the past 30 years, been distinguished by its transition from provision of essentially empirical treatments to a practice in which rational therapies, based on enhanced knowledge of the pathogenesis of disease, are administered. Coincidentally, the boundaries between organ specific specialties such as our own have become blurred. Thus, there is now great overlap with numerous other medical disciplines. These include: oncology, immunology / allergology, plastic surgery, oral medicine, and increasingly rheumatology and other disciplines in which management of chronic inflammatory disease plays an important role.

Dermatology is the branch of medicine dealing with the hair, nails, skin and its diseases. It is a specialty with both medical and surgical aspects. A Dermatologist treats diseases, in the widest sense, and some cosmetic problems of the skin, scalp, hair, and nails. Dermatology is the medical specialty which focuses on the diagnosis of skin diseases and disorders.

A physician who specialized in this field is called a dermatologist. Dermatologists identify and remove skin cancers, cysts, and other skin growths. They also help treat other skin issues such as acne, skin allergies, rashes, and abnormalities of the skin such as psoriasis, eczema, dandruff, dermatitis, and more. Many dermatologists also do some aesthetic, elective procedures such as laser treatments, anti-aging, Botox injections, and collagen injections.

The dataset contains 34 attributes, 33 of which are linear valued and one of them is nominal. The differential diagnosis of erythemato-squamous diseases is a real problem in dermatology. They all share the clinical features of erythema and scaling, with very little differences. The diseases in this group are psoriasis, seboreic dermatitis, lichen planus, pityriasis rosea, cronic dermatitis, and pityriasis rubra pilaris. Usually a biopsy is necessary for the diagnosis but unfortunately these diseases share many histopathological features as well. Another difficulty for the differential diagnosis is that a disease may show the features of another disease at the beginning stage and may have the characteristic features at the following stages. Patients were first evaluated clinically with 12 features. Afterwards, skin samples were taken for the evaluation of 22 histopathological features. The values of the histopathological features are determined by an analysis of the samples under a microscope.

Attribute Information: Clinical Attributes: (take values 0, 1, 2, 3,)

1: erythema

2: scaling

3: definite borders

4: itching

5: koebner phenomenon

6: polygonal papules

7: follicular papules

8: oral mucosal involvement

9: knee and elbow involvement

10: scalp involvement

11: family history, (0 or 1)

34: Age (linear)

Histopathological Attributes: (take values 0, 1, 2, 3)

12: melanin incontinence

13: eosinophils in the infiltrate

14: PNL infiltrate

15: fibrosis of the papillary dermis

16: exocytosis

17: acanthosis

18: hyperkeratosis

19: parakeratosis

20: clubbing of the rete ridges

21: elongation of the rete ridges

22: thinning of the suprapapillary epidermis

23: spongiform pustule

24: munro microabcess

25: focal hypergranulosis

26: disappearance of the granular layer

27: vacuolisation and damage of basal layer

28: spongiosis

29: saw-tooth appearance of retes

30: follicular horn plug

31: perifollicular parakeratosis

32: inflammatory monoluclear inflitrate

33: band-like infiltrate

1.2 Motivation

In recent years, data has become increasingly larger in both number of instances and the number of features with different ranges. This enormity may cause serious problems to the scalability and performance of many machine learning algorithms. Therefore, feature selection becomes very necessary for machine learning tasks when facing high dimensional data nowadays.

Feature selection is the process of selecting a subset of the features to describe a phenomenon from a larger set that may contain irrelevant or redundant features. In addition, reducing the number of features may help decrease the cost of acquiring data and might make the classification models easier to understand. Filters often give best results in terms of the final predictive accuracy for the calculation of Genetic Algorithm.

This thesis combines Ant Colony Optimization (ACO) algorithm and Genetic Algorithm (GA) to get the maximum performance of classifier.

1.3 Objectives of the Thesis

The objectives of the thesis are:

- To reduce the feature subsets using Ant Colony Optimization (ACO)
- To know the feature selection is importance for classifier
- To improve the performance of classifier
- To correctly classify unknown dataset with the reduced feature set

1.4 Organization of the Thesis

The body of the thesis consists of five chapters. In Chapter (1), the introduction, motivation of the thesis and objectives of the thesis are described. In Chapter (2), theory background of the system is described. Chapter (3) presents feature selection with ant colony algorithm and genetic algorithm. The corresponding design method and the implementation of the thesis are described in Chapter (4). The last Chapter (5) presents conclusion, limitation and further extension.

CHAPTER-2
THEORY BACKGROUND

This chapter describes the background theory of data mining, ant colony optimization and genetic algorithm.

2.1 Data Preprocessing

Real world database are highly susceptible to noisy, missing, and inconsistent data due to their typically huge size. The data can be preprocessed in order to help improve the quality of the data and consequently of the mining result.

Incomplete, noisy and inconsistent data are commonplace of large real world databases and data warehouses. Incomplete data can occur for a number of reasons. Attributes of interest may not always be available, such as customer information for sales transaction data. Other data may not be included simply because it was not considered important at the time of entry. Relevant data may not be recorded due to a misunderstanding, or because of equipment malfunctions. Data that were inconsistent with other recorded data may have been deleted. Furthermore, the recording of the history or modifications to the data may have been overlooked. Missing data, particularly for tuples with missing values for some attributes, may need to be inferred.

There are many possible reasons for noisy data (having incorrect attribute values). The data collection instruments used may be faulty. There may have been human or computer errors occurring at data entry. Errors in data transmission can also occur. Incorrect data may also result from inconsistencies in naming conventions or data codes used or inconsistent formats for input fields, such as date. Duplicate tuples also require data cleaning.

There are number of data preprocessing techniques. They are data cleaning, data integration, data transformation and data reduction.

2.1.1 Data Cleaning

Real-world data tend to be incomplete, noisy and inconsistent. Data cleaning can be applied to remove noise and correct inconsistencies in the data. Data cleaning routines tend to fill in missing values, smooth out noise while identifying or removing outliers and resolving inconsistencies.

2.1.1.1 Missing Value

The following methods can be used to fill the missing values.

- Ignore the tuple.
- Fill in the missing value manually.
- Use a global constant to fill in the missing value.
- Use the attribute mean to fill in the missing value.
- Use the attribute mean for all samples belonging to the same class as the given tuple.
- Use the most probable value to fill in the missing value.

2.1.1.2 Noisy Data

Noise is a random error or variance in a measured variable. The following data smoothing techniques can be used to remove:

- **Binning:** Binning methods smooth a sorted data value by consulting its "neighborhood", the values around it.
- **Regression:** Data can be smoothed by fitting the data to a function, such as with regression.

- **Clustering**: Outliers may be detected by clustering, where similar values are organized into groups, or "clusters". Values that outside of the set of clusters may be considered outliers.

2.1.2 Data Integration and Transformation

Data mining often requires data integration- the merging of data from multiple data stores. The data may also need to be transformed into forms appropriate for mining. Data analysis task will involve data integration, which combines data from multiple sources into a coherent data store. These sources may include multiple databases, data cubes or flat files.

2.1.2.1 Data Integration

In data integration, data analysis task will involve in data integration, which combines data from multiple sources into a coherent data store, as in data warehousing. These sources may include multiple databases, data cubes, or flat files.

There are a number of issues to consider during data integration. Schema integration and object matching can be tricky. Equivalent real-world entities from multiple data sources can be matched up as the entity identification problem. E.g. metadata for each attribute include the name, meaning, data type, and range of values permitted for the attributes, and null rules for handling blank, zero, or null values. Such metadata can help avoid errors in schema integration. The metadata may also be used to help transform the data.

Data integration is the detection and resolution of data value conflicts. For the same real world entity, attributes values from different sources may differ. This may be due to differences in representation,

scaling, or encoding. For instance, a weight attribute may be stored in metric units in one system and British imperial units in another. For a hotel chain, the price of rooms in different cities may involve not only different currencies but also different services and taxes.

2.1.2.2 Data Transformation

In data transformation, the data are transformed or consolidated into forms appropriate for mining. Data transformation can involve the following:

- Smoothing
- Aggregation
- Generalization
- Normalization
- Attribute construction

2.1.3 Data Reduction

Data reduction techniques can be applied to obtain a reduced representation of the data set that is much smaller in volume, yet closely maintains the integrity of the original data. Strategies for data reduction include the following:

- Data Cube aggregation
- Attribute subset selection
- Dimensionality reduction
- Discretization and concept hierarchy generation

scaling, or encoding. For instance, a weight attribute may be stored in metric units in one system and British imperial units in another. For a hotel chain, the price of rooms in different cities may involve not only different currencies but also different services and taxes.

2.1.2.2 Data Transformation

In data transformation, the data are transformed or consolidated into forms appropriate for mining. Data transformation can involve the following:

- Smoothing
- Aggregation
- Generalization
- Normalization
- Attribute construction

2.1.3 Data Reduction

Data reduction techniques can be applied to obtain a reduced representation of the data set that is much smaller in volume, yet closely maintains the integrity of the original data. Strategies for data reduction include the following:

- Data Cube aggregation
- Attribute subset selection
- Dimensionality reduction
- Discretization and concept hierarchy generation

2.2 Feature Selection

Feature selection process is very important which selects the informative features for used classification process. This is due to the fact that performance of the classifier is sensitive to the choice of the features used to construct the good classifier from small or high dimension data that are inherently noisy. Features selection methods aim at selecting a small or prespecified number of features leading to the best possible performance of the entire classifier [1].

Feature subset selection is a process of finding a subset of features that represents the full dataset from a much larger set. There may be thousands of features present in a real world datasets and each feature may carry only a little bit of information, it would be very difficult to treat all features. Therefore, it is very important to extract or select important features from the dataset.

There are many benefits of feature subset selection. It facilitates data visualization and provides better data understanding. It also reduces the complexity of training data that leads to reduced training times of the learning algorithm. Another very important factor of feature subset selection is to reduce the curse of dimensionality and improve the performance of prediction [5].

Categories of optimal feature subset selection are as follows:

Filter approaches

In the filter approach, no classification function is used and feature subsets are evaluated by other means. A type of filter approach, an exhaustive search is utilized to examine all the subsets of features. The method then identifies the subset with minimum number of features which classifies the training set with acceptable level of accuracy.

Undesirable attributes are filtered out of the data before classification begins. Filter approach attempts to remove irrelevant

features from feature set by score and ranking them based on certain statistical criteria and then features with lowest ranking values are removed from feature set [2].

Wrapper approach

In a wrapper approach, a classification function is used to evaluate the "goodness" of the feature subsets developed. The feature subset selection algorithm is wrapped around the classification function. If decision trees use n of the N total features, all feature subsets which have all of these n features will create the same tree with the same accuracy.

Feature selection is "wrapped" in a learning algorithm. It generates various subsets of features and then a specific classification is applied to evaluate accuracy of these subsets. Wrapper methods can broadly be classified into two categories: greedy methods and randomized methods.

Wrapper methods are successful in feature selection, there may be computationally expensive, because they require the retraining of a classifier on data with a large number of features. The task of feature selection is to choose a subsets **S** of the input variables that maximizes the performance of the classifier on the test set.

2.3 Supervised Learning and Unsupervised Learning

Supervised Learning is a machine learning paradigm for deducing a function from training data. The training data consist of pairs of input objects and desire outputs. The output of the function can be a continuous value (called regression), or can predict a class label of the input object (call classification). The task of the supervised learner is to predict the value of the function for any valid input object after having seen a number of training examples (i.e. pairs of input and target output).

In unsupervised methods, no target variable is identified as such. Unsupervised learning in which the class labels of each training tuple is not known, and the number or set of classes to be learned may not be known in advance. The lack of knowledge of the cluster membership for every instance in the data clustering can be also referred to as unsupervised learning.

2.4 Classification

Classification is the grouping of things according to the characteristics. Data Classification is two-step processes. In the first step, a classifier is built describing a predetermined set of data classes or concepts. A classification algorithm builds the classifier by analyzing or "learning from" a training set made up of database tuples and their associated class labels [4].

Each tuple is assumed to belong to a predefined class as determined by another database attribute called the class label attribute. The class label attribute is discrete-valued and unordered. The individual tuples making up the training set are referred to as training tuples and are selected from the database under analysis. Classification may indicate similarity to objects that are definitely members of a given class. Classification consists of predicting a certain outcome based on a given input. In order to predict the outcome, the algorithm processes a training set containing a set of attributes and the respective outcome, usually called goal or prediction attribute.

In the second step, the model is used for classification. The predictive accuracy of the classifier is estimated. It is likely to use the training set to measure the accuracy of the classifier, this estimate would likely be optimistic, because the classifier tends to over fit the data. A test

by ovals. Some decision tree algorithms produce only binary trees, whereas others can produce non-binary trees.

The construction of decision tree classifiers does not require any domain knowledge or parameter setting, and is appropriate for exploratory knowledge discovery. Decision trees can handle high dimensional data. The learning and classification steps of decision tree induction are simple and fast. In general, decision tree classifiers have good accuracy. However, successful use may depend on the data at hand.

Decision tree induction algorithms have been used for classification in many application areas, such as medicine, manufacturing and production, financial analysis, astronomy, and molecular biology. Decision trees are the basis of several commercial rule induction systems.

2.4.2 Bayesian Classification

Bayesian classifiers are statistical classifiers. They can predict class membership probabilities, such as the probability that a given tuple belongs to a particular class. Bayesian classification is based on Bayes' theorem. Studies comparing classification algorithms have found a simple Bayesian classifier known as the Naive Bayesian classifier to be comparable in performance with decision tree and selected neural network classifiers. Bayesian classifiers have also exhibited high accuracy and speed when applied to large databases.

Naive Bayesian classifiers assume that the effect of an attribute value on a given class is independent of the values of the other attributes. This assumption is called class conditional independence. It is made to simplify the computations involved and is considered "naive". Bayesian belief networks are graphical models, which unlike naïve Bayesian

by ovals. Some decision tree algorithms produce only binary trees, whereas others can produce non-binary trees.

The construction of decision tree classifiers does not require any domain knowledge or parameter setting, and is appropriate for exploratory knowledge discovery. Decision trees can handle high dimensional data. The learning and classification steps of decision tree induction are simple and fast. In general, decision tree classifiers have good accuracy. However, successful use may depend on the data at hand.

Decision tree induction algorithms have been used for classification in many application areas, such as medicine, manufacturing and production, financial analysis, astronomy, and molecular biology. Decision trees are the basis of several commercial rule induction systems.

2.4.2 Bayesian Classification

Bayesian classifiers are statistical classifiers. They can predict class membership probabilities, such as the probability that a given tuple belongs to a particular class. Bayesian classification is based on Bayes' theorem. Studies comparing classification algorithms have found a simple Bayesian classifier known as the Naive Bayesian classifier to be comparable in performance with decision tree and selected neural network classifiers. Bayesian classifiers have also exhibited high accuracy and speed when applied to large databases.

Naive Bayesian classifiers assume that the effect of an attribute value on a given class is independent of the values of the other attributes. This assumption is called class conditional independence. It is made to simplify the computations involved and is considered "naive". Bayesian belief networks are graphical models, which unlike naïve Bayesian

classifier; allow the representation of independencies among subsets of attributes. Bayesian belief networks can also be used for classification.

2.4.3 Classification by Back propagation

Back propagation is a neural network learning algorithm. The field of neural networks was originally kindled by psychologists and neurobiologists who sought to develop and test computational analogues of neurons. A neural network is a set of connected input/output units in which each connection has a weights associated with it. During the learning phase, the network learns by adjusting the weights so as to be able to predict the correct class label of the input tuples. Neural network learning is also referred to as connectionist learning due to the connections between units.

Neural networks involve long training times and are more suitable for application where this is feasible. They require a number of parameters that are typically best determined empirically, such as the network topology or "structure". Neural networks have been criticized for their poor interpretability.

There are many different kinds of neural networks and neural network algorithms. The most popular neural network algorithm is back propagation.

The back propagation algorithm performs learning on a multilayer feed-forward neural network. It iteratively learns a set of weights for prediction of the class label of tuples. A multilayer feed-forward neural network consists of an input layer, one or more hidden layers and an output layer.

Each layer is made up of units. The inputs to the network correspond to the attributes measured for each training tuple. The inputs

are fed simultaneously into the units making up the input layer. These input pass through the input layer and are then weighted and fed simultaneously to a second layer of "neuron like" units, known as hidden layer. The output of the hidden layer units can be input to another hidden layer and so on. The number of hidden layers is arbitrary, although in practice, usually only one is used. The weighted outputs of the last hidden layer are input to units making up the output layer, which emits the network's prediction for given tuples.

2.4.4 Classifier Accuracy Measures

Using training data to derive a classifier or predictor and then to estimate the accuracy of the resulting learned model can result in misleading overoptimistic estimates due to overspecialization of the learning algorithm to the data.

Instead, accuracy is better measured on a test set consisting of class-labeled tuples that were not used to train the model. The accuracy of a classifier on a given test set is the percentage of test set tuples that are correctly classified by the classifier. In the pattern recognition literature, this is also referred to as the overall recognition rate of the classifier, that is, it reflects how well the classifier recognizes tuples of the various classes.

The sensitivity and specificity measures can be used. Sensitivity is also referred to as the true positive (recognition) rate (that is, the proportion of positive tuples that are correctly identified), while specificity is the true negative rate.

2.4.5 Cross-Validation

In 10-fold cross-validation, the initial data are randomly partitioned into k mutually exclusive subsets or "folds," D1, D2, … , D10 , each of approximately equal size. Training and testing is performed 10 times. In iteration i, partition Di is reserved as the test set, and the remaining partitions are collectively used to train the model. That is, in the first iteration, subsets D2, … , D10 collectively serve as the training set in order to obtain a first model, which is tested on D1; the second iteration is trained on subsets D1, D3, … , D10 and tested on D2; and so on.

In general, stratified 10-fold cross-validation is recommended for estimating accuracy (even if computation power allows using more folds) due to its relatively low bias and variance.

2.5 Introduction to Ant Colony Optimization (ACO) System

Ant colony optimization is a technique for optimization that was introduced in the early 1990's. The inspiring source of ant colony optimization is the foraging behavior of real ant colonies. This behavior is exploited in artificial ant colonies for the search of approximate solutions to discrete optimization problems, to continuous optimization problems, and to important problems in telecommunications, such as routing and load balancing.

Several different aspects of the behavior of ant colonies have inspired different kinds of ant algorithms. Examples are foraging, division of labor, brood sorting, and cooperative transport.

A foraging ant deposits a chemical on the ground which increases the probability that other ants will follow the same path. In fact, an important insight of early research on ants' behavior was that most of the communication among individuals, or between individuals and the

environment is based on the use of chemicals produced by the ants. These chemicals are called pheromones. Particularly important for the social life of some ant species is the trail pheromone. Trail pheromone is a specific type of pheromone that some ant species use for marking paths on the ground, for example, paths from food sources to the nest. By sensing pheromone trails foragers can follow the path to food discovered by other ants. This collective trail-laying and trail-following behavior whereby an ant is influenced by a chemical trail left by other ants is the inspiring source of ACO [6].

The foraging behavior of many ant species is based on indirect communication mediated by pheromones. While walking from food sources to the nest and vice versa, ants deposit pheromones on the ground, forming in this way a pheromone trail. Ants can smell the pheromone and they tend to choose, probabilistically, paths marked by strong pheromone concentrations. For example, we consider two paths of route .First, one route of two branches has the equal lengths. At the start, ants were left free to move between the nest and the food source and the percentage of ants that chose one or the other of the two branches were observed over time. Although in the initial phase random choices occurred, eventually all the ants used the same branch. When a trial starts there is no pheromone on the two branches. Hence, the ants do not have a preference and they select with the same probability any of the branches. Because ants deposit pheromone while walking, a larger number of ants on a branch results in a larger amount of pheromone on that branch; this larger amount of pheromone in turn stimulates more ants to choose that branch again, and so on until finally the ants converge to one single path.

In the second, the route of the two branches has the different length. The long branch was twice as long as the short one. In this case, in most of the trials, after some time all the ants chose to use only the short

branch. As in the first experiment, ants leave the nest to explore the environment and arrive at a decision point where they have to choose one of the two branches. Because the two branches initially appear identical to the ants, they choose randomly.

ACO algorithms can be used to solve both static and dynamic combinatorial optimization problems. Static problems are those in which the characteristics of the problem are given once and for all when the problem is defined and did not changed while the problem is being solved. The following are the extension of the Ant Colony Optimization (ACO):

- Ant System
- Elitist Ant System
- Rank-based Ant System
- MIN-MAX Ant System
- Ant Colony System

2.5.1 Ant System

The two main phases of the AS algorithm constitute the ants' solution construction and the pheromone update. In AS a good heuristic to initialize the pheromone trails is to set them to a value slightly higher than the expected amount of pheromone deposited by the ants in one iteration. A rough estimate of this value can be obtained by setting, $\tau_{ij} = \tau_0 = m/C^{nn}$, where m is the number of ants, and C^{nn} is the length of a tour generated by the nearest-neighbor heuristic. The reason for this choice is that if the initial pheromone valuest0's are too low, then the search is quickly biased by the first tours generated by the ants, which in general leads toward the exploration of inferior zones of the search space. On the other side, if the initial pheromone values are too high, then

$$\tau_{ij} = \rho * \tau_{ij} + \sum_{r=1}^{w-1} (w - r)\Delta\tau_{ij}{}^{r} + w\Delta\tau_{ij}{}^{bs}$$

where,

$$\Delta\tau_{ij}^{r}=1/C^{r} \text{ and } \Delta\tau_{ij}^{bs}=1/C^{bs}$$

AS_{rank} performs slightly better than EAS and significantly better than AS.

2.5.4 Min-Max Ant System

MAX–MIN Ant System (MMAS) introduces four main modifications with respect to AS. First, it strongly exploits the best tours found: only either the iteration-best ant, that is, the ant that produced the best tour in the current iteration, or the best-so-far ant is allowed to deposit pheromone. Unfortunately, such a strategy may lead to a stagnation situation in which all the ants follow the same tour, because of the excessive growth of pheromone trails on arcs of a good, although suboptimal, tour. A second modification introduced by MMAS is that it limits the possible range of pheromone trail values to the interval $\{\tau_{max}, \tau_{min}\}$.

Third, the pheromone trails are initialized to the upper pheromone trail limit, which, together with a small pheromone evaporation rate, increases the exploration of tours at the start of the search. Finally, in MMAS, pheromone trails are reinitialized each time the system approaches stagnation or when no improved tour has been generated for a certain number of consecutive iterations.

2.5.5 Ant Colony System

First, it exploits the search experience accumulated by the ants more strongly than AS does through the use of a more aggressive action choice rule. Second, pheromone evaporation and pheromone deposit take place only on the arcs belonging to the best-so-far tour. Third, each time an ant uses an arc (i,j) move from city i to city j, it removes some pheromone from the arc to increase the exploration of alternative paths.

Artificial ants iteratively sample tours through a loop that includes a tour construction biased by the artificial pheromone trails and the heuristic information. The main mechanism at work in ACO algorithms that triggers the discovery of good tours is the positive feedback given through the pheromone update by the ants: the shorter the ant's tour, the higher the amount of pheromone the ant deposits on the arcs of its tour.

This in turn leads to the fact that these arcs have a higher probability of being selected in the subsequent iterations of the algorithm. The emergence of arcs with high pheromone values is further reinforced by the pheromone trail evaporation that avoids an unlimited accumulation of pheromones and quickly decreases the pheromone level on arcs that only very rarely, or never, receives additional pheromone.

2.6 Genetic Algorithm

Genetic algorithms attempt to incorporate ideas of natural evolution. In general, genetic learning starts as follows. An initial population is created consisting of randomly generated rules. Each rule can be represented by a string of bits. As a simple example, suppose that samples in a given training set are described by two Boolean attributes, A1 and A2, and that there are two classes, C1 and C2. The rule "IF A1 AND NOT A2 THEN C2 " can be encoded as the bit string "100," where

the two leftmost bits represent attributes A1 and A2 , respectively, and the rightmost bit represents the class. Similarly, the rule "IF NOT A1 AND NOT A2 THEN C1" can be encoded as "001." If an attribute has k values, where k > 2, then k bits may be used to encode the attribute's values. Classes can be encoded in a similar fashion.

Based on the notion of survival of the fittest, a new population is formed to consist of the fittest rules in the current population, as well as offspring of these rules. Typically, the fitness of a rule is assessed by its classification accuracy on a set of training samples.

Offspring are created by applying genetic operators such as crossover and mutation. In crossover, substrings from pairs of rules are swapped to form new pairs of rules. In mutation, randomly selected bits in a rule's string are inverted [11].

The process of generating new populations based on prior populations of rules continues until a population, P, evolves where each rule in P satisfies a prespecified fitness threshold. Genetic algorithms are easily parallelizable and have been used for classification as well as other optimization problems. In data mining, they may be used to evaluate the fitness of other algorithms.

The following is genetic algorithm template:

```
Choose an initial population of chromosomes;
while termination condition not satisfied do
        repeat
          if crossover condition satisfied then
           {select parent chromosomes;
            choose crossover parameters;
            perform crossover parameters};
          if mutation condition satisfied then
           {choose mutation points;
            perform mutation};
          evaluate fitness of offspring
        until sufficient offspring created;
select new population;
end while
```

Figure (2.1) A Genetic Algorithm Template.

Genetic algorithm works with a set of individuals, representing possible solutions of the task. The selection principle is applied by using a criterion, giving an evaluation for the individual with respect to the desired solution. The best-suited individuals create the next generation.

For the genetic algorithms, the chromosomes represent set of genes, which code the independent variables. Every chromosome represents a solution of the given problem. Individual and vector of variables will be used as other words for chromosomes. From other hand, the genes could be Boolean, integers, floating point or string variables as well as any combination of the above. A set of different chromosomes forms a generation. By means of evolutionary operators, like selection, recombination and mutation an offspring population is created.

Initialization

Initially, many individual solutions are randomly generated to form an initial population. The population size depends on the nature of the problem, but typically contains several hundred or thousands of possible solutions. Traditionally, the population is generated randomly, allowing the entire range of possible solutions. Occasionally, the solution may be "seeded" in areas where optimal solutions are likely to be found.

Finally, as to how the population is chosen, it is nearly always assumed that initialization should be random. Some reports have found that including a high-quality solution, obtained from another technique, can help a GA find better solutions rather more quickly than it can from a random start. However, there is also the possibility of inducing premature convergence.

Selection

In the nature, the selection of individuals is performed by survival of the fittest. The more one individual is adapted to the environment: the bigger are its chances to survive and create an offspring and thus transfer its genes to the next population. During each successive generation, a proportion of the existing population is selected to breed a new generation. Individual solutions are selected through a fitness-based process, where fitter solutions are typically more likely to be selected. Certain selection methods rate the fitness of each solution and preferentially select the best solutions. Other methods rate only a random sample of the population, as the former process may be very time consuming.

The fitness function is defined over the genetic representation and measures the quality of the represented solution. The fitness function is always problem dependent. In some problem, it is hard or even impossible to define the fitness expression; in this case, a simulation may

be used to determine the fitness function value of a phenotype or even interactive genetic algorithms are used. The selection of individuals is based on an evaluation of fitness function or fitness functions. If the optimization problem is a minimization one, than individuals with small value of the fitness function will have bigger chances for recombination and respectively for generating offspring.

Crossover

Crossover keeps useful informative blocks and produces offspring which have the same distribution than the parents. Off-springs are kept, only if they fit better than the least good individual of the population. According to generate new offspring, it is also possible to use crossover or mutation. There are many examples of both in the literature. The first strategy initially tries to carry out crossover, then attempts mutation on the off-spring. It is conceivable that in some cases nothing actually happens at all with this strategy- the off-spring are simply clones of the parent. Others always do something, either crossover of mutation, but not both [14].

After selection, individuals from the mating pool are recombined (or crossed over) to create new, hopefully better, offspring. Many of the recombination operators used in the literature are problem-specific and in this section we will introduce a few generic (problem independent) crossover operators. It should be noted that while for hard search problems, many of the following operators are not scalable, they are very useful as a first option. In most recombination operators, two individuals are randomly selected and are recombined with a probability, called the crossover probability. That is, a uniform random number **r** is generated.

Mutation

If we use a crossover operator, such as one-point crossover, we may get better and better chromosomes but the problem is, if the two

parents (or worse, the entire population) has the same allele at a given gene then one-point crossover will not change that. In other words, that gene will have the same allele forever. Mutation is designed to overcome this problem in order to add diversity to the population and ensure that it is possible to explore the entire search space. In evolutionary strategies, mutation is the primary variation/search operator. An introduction to evolutionary strategies sees. Unlike evolutionary strategies, mutation is often the secondary operator in GAs, performed with a low probability. One of the most common mutations is the bit-flip mutation. In bitwise mutation, each bit in a binary string is changed (a 0 is converted to 1, and vice versa) with a certain probability, known as the mutation probability. As mentioned earlier, mutation performs a random walk in the vicinity of the individual. Other mutation operators, such as problem-specific ones, can also be developed and are often used in the literature.

Once the new offspring solutions are created using crossover and mutation, we need to introduce them into the parental population. There are many ways we can approach this. Bear in mind that the parent chromosomes have already been selected according to their fitness, so we are hoping that the children (which include parents which did not undergo crossover) are among the fittest in the population and so we would hope that the population will gradually, on average, increase its fitness. Some of the most common replacement techniques are outlined below.

Delete-all: This technique deletes all the members of the current population and replaces them with the same number of chromosomes that have just been created. This is probably the most common technique and will be the technique of choice for most people due to its relative ease of implementation. It is also parameter-free, which is not the case for some other methods.

Steady-state: This technique deletes n old members and replaces them with n new members. The number to delete and replace n, at any one time is a parameter to this deletion technique. Another consideration for this technique is deciding which members to delete from the current population. Do you delete the worst individuals, pick them at random or delete the chromosomes that you used as parents? Again, this is a parameter to this technique.

Steady-state-no-duplicates: This is the same as the steady-state technique but the algorithm checks that no duplicate chromosomes are added to the population. This adds to the computational overhead but can mean that more of the search space is explored.

CHAPTER-3

CLASSIFICATION WITH FEATURE SELECTION

This chapter describes the preprocessing method, the filter approach by using Ant Colony Optimization (ACO) and classification with Genetic Algorithm and performance evaluation. The data can be preprocessed to improve the efficiency and ease of the mining process. Ant Colony Optimization (ACO) method generates the optimal features of the data in dermatology data set and reducing the number of features required for learning the classification rules. The filer method selects a good subset of the feature. The Genetic Algorithm improved the results by producing a higher accuracy. Performance evaluation of the classifier is evaluated by the 10-fold cross-validation.

3.1 Attribute Mean to Fill in the Missing Value

In mathematics, the "mean" is a kind of average found by dividing the sum of a set of numbers by the count of numbers in the set.

For missing values, this thesis uses the attribute mean for all samples belonging to the same class as the given tuple. Let $x_1, x_2, .., x_N$ be the set of N values, such as for attributes.

$$\bar{x} = \frac{\sum_{i=1}^{N} x_i}{N} = \frac{x_1 + x_2 + \cdots + x_N}{N} \tag{3.1}$$

3.2 Ant Colony Optimization

ACO has been successfully applied in the selection of optimal minimal subset of feature. The following is the pseudo code for ACO:

```
Procedure ACO Metaheuristic Static
    Schedule Activities
        ConstructAntsSolutions
        UpdatePheromones
        DaemonActions %optional
    end-ScheduleActivities
end-procedure
```

Figure 3.1 Pseudo Code for Ant Colony Optimization (ACO) Algorithm

ACO algorithms can be used in any optimization problem, if the following aspects are provided:

(1) Graph construction

(2) Solution construction

(3) Pheromone trails and heuristic information

(4) Pheromone update rule

Graph Construction: The problem needs to be represented in a shape of graph, i.e. nodes represent features, and the edges between them denote the choice of the next feature. Nodes are fully connected. The search for the optimal feature subset is an ant traversal through the graph in which the minimum number of nodes is visited.

Solution construction: Each ant starts randomly chosen start feature and in each iteration adds an unvisited feature to its partial tour. At each step, each ant constructs a solution. Then each ant will share its solution feedback with the entire colony by updating a global data structure called the pheromone matrix. This data structure simulates the pheromone trails. Each entry in the pheromone matrix shows the desirability of each solution component. At the end of each iteration, the pheromone associated with each solution component is reinforced based on the quality of the solution that comprises the particular solution component.

In subsequent iterations, ants will use the pheromone intensities of available solution components to guide solution construction. As a result of repeated pheromone reinforcements, a subset of solution components will emerge to have pheromone intensities much higher than the others. At this point, ants will construct identical or nearly identical solutions using these highly desirable components. The solution construction terminates when it can satisfy the stopping criterion (e.g. suitably high classification accuracy has been achieved).

3.2.1 Initialization of Pheromone Values

The presence of pheromone values on the edges is the basic component of the ACO. Initially, it is initialized by some small random value. In our experiments, the pheromone values on all edges are initialized at the start of the algorithm with the same amount of pheromone. In this way, no attribute is preferred over other attributes by the first ant. The initial pheromone is calculated according to the following Equation (3.2.1):

$$\tau_{ij}(t = 1) = \frac{1}{N} \tag{3.2}$$

Where N is the total number of attributes present in the dataset excluding the class attribute. An ant starts its journey from the initial point. Suppose that first it goes to the attribute A1, and then it visits A2, A3 and then selects the sink node. The tour is terminated as soon as the sink node is selected [9].

3.2.2 Heuristic Information Value of an Attribute

Heuristic function indicates the quality of an attribute. Its value greatly influences an ant's decision to move from one node to another. A

good heuristic function is very helpful in solving problems by ACO. Heuristic value helps in finding a good next step in constructing a solution. F-score is used as a measurement to determine the importance of feature. After computing the F-score for each feature in the dataset, average F-score will be computed and it will considered as threshold for choosing feature in the feature subset. Feature with F-score equal to or greater to the threshold will be chosen. In feature selection, F-score is used as a measurement to determine the feature importance. This measurement is used to judge the favoritism capability of a feature. High value of F-score indicates favorable feature.

Let S be a set of given N features, and ()f τ be the pheromone level (desirability measure) of feature f to be in the selected subset of features s where s S $\subset$. Initially, the desirability of each feature will be the same, but the desirability of those features which are more important, increases in each step.

Here we used F-score as heuristic information. Eq.(1) defines the F-score of the i th feature.

$$\eta_i = \frac{\sum_{c=1}^{v}\left(\bar{x}^{(c)}-\bar{x}_i\right)^2}{\sum_{c=1}^{v}\left\{\frac{1}{N^{(c)}-1}\sum_{j=1}^{N_i^{(c)}}\left(x_{i,j}^{(c)}-\bar{x}_i^{(c)}\right)^2\right\}} \tag{3.3}$$

where i$\in\{1, 2, . .,N_F\}$.

A larger F-score corresponds to a greater likelihood that this feature is discriminative where v is the number of categories of target variable; N_F the number of features; $N_i^{(c)}$the number of samples of the I th feature with categorical value c, c$\in\{1,2,3,...,v\}$, i$\in\{1, 2, . ., N_F\}$; $x_{i,j}^{(c)}$ the j th training sample for the i th feature with categorical value c , j$\in\{1, 2, . ., N_i^{(c)}\}$; $\bar{x}_i$ the mean of the i th feature; $x_i^{(c)}$ the mean of the i th feature with categorical value c.

3.2.3 Selection of an Attribute

An ant uses two components to calculate the probability of moving from the present node to the next node. The first component is the amount of pheromone present on the edge between node i and node j and second is the heuristic value that describes the worth of a node. The probability with which an ant chooses node j as the next node, after it has arrived at node i. Node j has to be in the set S of nodes that have not been visited.

The goal of this optimization algorithm is to minimize the classification error in predicting the output. In this hybrid approach, the role of each ant is to build a solution subset. The ''ants'' build solutions applying a probabilistic decision policy to choose next node. In this case each subset of feature represents a state. The state transition rules help ants to select features using the pheromone trail and the heuristic value. Each ant chooses a particular feature by maximizing a product of these two parameters.

An ant chooses a feature as follows:

$$P_i^{\,k}(t) = \begin{cases} \dfrac{[\tau_i(t)]^{\alpha}.[\eta_i(t)]^{\beta}}{\sum_{u \epsilon J^k}[\tau_u(t)]^{\alpha}.[\eta_u(t)]^{\beta}}, & if\ i \epsilon J^k \\ 0 & , otherwise \end{cases} \qquad (3.4)$$

Where J^k is the set of feasible features that can be added to the partial solution; 0 $\alpha \geq$ and 0 $\beta \geq$ are two parameters that determine the importance of the pheromone value and heuristic desirability.

The parameters α and β are influencing factors of pheromone value and heuristic value respectively. Where, $J^k$is the set of feasible features that can be added to the partial solution 0 $\alpha \geq$ and 0 $\beta \geq$ are two parameters that determine the importance of the pheromone value and heuristic desirability. Once complete solutions have been built ,

pheromone trails are updated. First, the pheromone is evaporated on all arcs and then all ants deposit pheromone on the arcs which are part of the solutions they have just computed. Evaporation of the pheromone trails is included to help ants "forget" bad solutions that were learned early on in the algorithm run.

3.2.4 Pheromone Updating

The pheromone values are updated after each ant completes its tour so that future ants can make use of this information in their search. The amount of pheromone on each link occurring in the current feature subset selected by an ant is updated.

InAS_{rank}, which is used in this research, each ant deposits an amount of pheromone that decreases with its rank. Additionally, the best-so-far ant always deposits the largest amounts of pheromone in each iteration.

Before updating the pheromone trails, the ants are sorted by increasing tour length and the quantity of pheromone an ant deposits is weighted according to the rank r of the ant. In each iteration, only the $(W-1)$ best ranked ants and the ant that produced the best-so-far tour are allowed to deposit pheromone. The best-so-far tour gives the strongest feedback, with weigh W. The r^{th} best ant of the current iteration contributes to pheromone updating with the value $1/C^r$ multiplied by a weight given by max $\{0, W-1\}$. Thus, the AS_{rank} pheromone update rule is

The best-so-far tour gives the strongest feedback, with weight w (i.e. its contribution $\Delta\tau_i^{bs}$ is multiplied by w); the r -th best ant of the current iteration contributes to pheromone updating with the value $\Delta\tau_i^r$

multiplied by a weight given by max{0,w-r }and finally ρ is an amount of evaporation. Thus, the AS_{rank} pheromone update rule is:

$$\tau_{ij} = \rho * \tau_{ij} + \sum_{r=1}^{w-1}(w-r)\Delta\tau_{ij}{}^{r} + w\Delta\tau_{ij}{}^{bs} \quad (3.5)$$

Where $\Delta\tau_{ij}^{r} = 1/C^{r}$, $\Delta\tau_{ij}^{bs} = 1/C^{bs}$

The pheromone is updated according to both the measure of the goodness of the ant's feature subset γ 'and the size of the subset itself [7].

Where w is weight parameter that is varied by the r^{th} ranks except for theT^{bs}, the rank of each timetable is considered by using its total violation index (Z). A number of ranks(r) selected for update are calculated from w-1. For timetable with the highest rank (r=1) and the lowest Z in current iteration, more pheromone value is added than in other ranks.

3.3 Filter for Feature Selection

Feature selection methods search through the subsets of features and try to find the best one among the competing candidate subsets according to some evaluation function.

Feature selection can be classified into two categories based on whether or not feature selection is done independent of learning algorithm used to construct the classifier. If feature selection is done independent of the learning algorithm, the technique is said to follow a filter approach.

Filter approach attempts to remove irrelevant features from features from feature set by score and ranking them based on certain statistical criteria and then features with lowest ranking values are removed from feature set. Feature selection is performed only once, and then different classifiers can be used.

3.4 Genetic Algorithm

A genetic algorithm is a search technique used in computing to find exact or approximate solutions to optimization and search problems. Genetic algorithms are a particular class of evolutionary algorithms (EA) that use techniques inspired by evolutionary biology such as inheritance, mutation, selection and crossover.

Genetic algorithm works with a set of individuals, representing possible solutions of the task. The selection principle is applied by using a criterion, giving an evaluation for the individual with respect to the desired solution. The best-suited individuals create the next generation.

For the genetic algorithms, the chromosomes represent set of genes, which code the independent variables. Every chromosome represents a solution of the given problem. Individual and vector of variables will be used as other words for chromosomes. From other hand, the genes could be Boolean, integers, floating point or string variables as well as any combination of the above. A set of different chromosomes forms a generation. By means of evolutionary operators, like selection, recombination and mutation an offspring population is created [13].

3.4.1 Initialization

Initially, many individual solutions are randomly generated to form an initial population. The population size depends on the nature of the problem, but typically contains several hundred or thousands of possible solutions. Traditionally, the population is generated randomly, allowing the entire range of possible solutions. Occasionally, the solution may be "seeded" in areas where optimal solutions are likely to be found.

the off-spring. It is conceivable that in some cases nothing actually happens at all with this strategy- the off-springs are simply clones of the parent. Others always do something, either crossover of mutation, but not both.

After selection, individuals from the mating pool are recombined (or crossed over) to create new, hopefully better, offspring. Many of the recombination operators used in the literature are problem-specific and in this section we will introduce a few generic (problem independent) crossover operators. It should be noted that while for hard search problems, many of the following operators are not scalable, they are very useful as a first option. In most recombination operators, two individuals are randomly selected and are recombined with a probability, called the crossover probability. That is, a uniform random number **r**, is generated.

If we use a crossover operator, such as one-point crossover, we may get better and better chromosomes but the problem is, if the two parents (or worse, the entire population) has the same allele at a given gene then one-point crossover will not change that. In other words, that gene will have the same allele forever. Mutation is designed to overcome this problem in order to add diversity to the population and ensure that it is possible to explore the entire search space. One of the most common mutations is the bit-flip mutation. In bitwise mutation, each bit in a binary string is changed (a 0 is converted to 1, and vice versa) with a certain probability, known as the mutation probability.

Once the new offspring solutions are created using crossover and mutation, we need to introduce them into the parental population. There are many ways we can approach this. Bear in mind that the parent chromosomes have already been selected according to their fitness, so we are hoping that the children (which include parents which did not undergo

the off-spring. It is conceivable that in some cases nothing actually happens at all with this strategy- the off-springs are simply clones of the parent. Others always do something, either crossover of mutation, but not both.

After selection, individuals from the mating pool are recombined (or crossed over) to create new, hopefully better, offspring. Many of the recombination operators used in the literature are problem-specific and in this section we will introduce a few generic (problem independent) crossover operators. It should be noted that while for hard search problems, many of the following operators are not scalable, they are very useful as a first option. In most recombination operators, two individuals are randomly selected and are recombined with a probability, called the crossover probability. That is, a uniform random number **r**, is generated.

If we use a crossover operator, such as one-point crossover, we may get better and better chromosomes but the problem is, if the two parents (or worse, the entire population) has the same allele at a given gene then one-point crossover will not change that. In other words, that gene will have the same allele forever. Mutation is designed to overcome this problem in order to add diversity to the population and ensure that it is possible to explore the entire search space. One of the most common mutations is the bit-flip mutation. In bitwise mutation, each bit in a binary string is changed (a 0 is converted to 1, and vice versa) with a certain probability, known as the mutation probability.

Once the new offspring solutions are created using crossover and mutation, we need to introduce them into the parental population. There are many ways we can approach this. Bear in mind that the parent chromosomes have already been selected according to their fitness, so we are hoping that the children (which include parents which did not undergo

crossover) are among the fittest in the population and so we would hope that the population will gradually, on average, increase its fitness.

Crossover operator is mainly responsible for the search aspect of genetic algorithms, even though mutation operator is also used for this purpose sparingly. Mutation operator changes a 1 to 0 and vice versa with a small mutation probability (p_m). The need for mutation is to keep diversity in the population. It alerts each bit randomly with a small mutation probability (p_m) with a typical value of less than 0.1.

The choice of mutation probability (p_m) and crossover probability (p_c) as the control parameters can be a complex nonlinear optimization problem to solve. Furthermore, their settings are critically dependent upon the nature of the objective function. This selection issue still remains open to suggestion although some guidelines have been introduced.

Fitness Computation

For the fitness computation, the following procedure is executed.

- Suppose there are N number of features present in a particular chromosome (i.e., there are total N number of 1's in that chromosome).
- Construct a classifier with only these N features.
- Here, initially the training data is divided into 3 parts. The above classifier is trained using 2/3 of the training set with the features encoded in that chromosome and tested with the remaining 1/3 part.
- Now, the overall F-measure value of this classifier for the 1/3 training data is calculated.
- Steps 2 and 3 are repeated 3 times to perform 3-fold cross validation.

- The average F-measure value of this 3-fold cross validation is used as the fitness value of the particular chromosome. The objective is to maximize this fitness value using the search capability of GA [12].

3.4.4 Termination Condition

In this approach, the processes of fitness computation, selection, crossover, and mutation are executed for a maximum number of generations. The best string seen up to the last generation provides the solution to the above feature selection problem. Elitism is implemented at each generation by preserving the best string seen up to that generation in a location outside the population. Thus on termination, this location contains the best feature combination.

The general process is repeated until a termination condition has been reached. Common terminated conditions are:

- A solution is found to satisfy minimum criteria
- Fixed number of generations reached
- Allocated budget (computation time or money) reached
- The highest ranking solution's fitness is reaching or has reached a plateau such that successive iterations no longer produce better results
- Manual inspection
- Combinations of the above

3.5 Performance Evaluation

Measure of the performance evaluation depends on the learning task. If the task is diagnosis or prognosis, classification accuracy is the

most frequently used quality evaluation measure, in addition to the interpretability of results.

The classification accuracy is measured by using sensitivity and specificity. Sensitivity measures the fraction of positive cases that are classified as positives. Specificity measures the fraction of negative cases classified as negatives.

$$sensitivity = \frac{t-pos}{pos} \tag{3.6}$$

$$specificity = \frac{t-neg}{neg} \tag{3.7}$$

$$precision = \frac{t\_pos}{t\_pos+f\_pos} \tag{3.8}$$

where t pos is the number of true positives ("cancer" tuples that were correctly classified such as), pos is the number of positive ("cancer") tuples, t neg is the number of true negatives ("not cancer" tuples that were correctly classified as such), neg is the number of negative ("not cancer") tuples, and f pos is the number of false positives ("not cancer" tuples that were incorrectly labeled as "cancer"). It can be shown that accuracy is a function of sensitivity and specificity:

$$A = sensitivity\frac{pos}{(pos+neg)} + specificity\frac{neg}{(pos+neg)} \tag{3.9}$$

The true positives, true negatives, false positives, and false negatives are also useful in assessing the costs and benefits (or risks and gains) associated with a classification model. The cost associated with a false negative (such as, incorrectly predicting that a cancerous patient is not cancerous) is far greater than that of a false positive (incorrectly yet conservatively labeling a noncancerous patient as cancerous).

3.5.1 10-Fold Cross-Validation

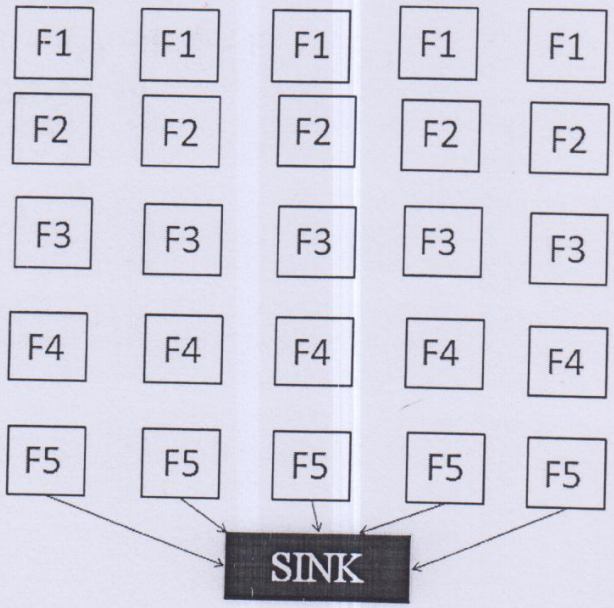
For 10-fold classification, the dataset is split into the number 10 entered by the user. The entire process from creating training set and test set to calculating the accuracy is performed 10 times using each set as the test set in each iteration. The training set is formed by merging the remaining 9 sets. The accuracies obtained from all iterations are averaged to get the accuracy of the classifier.

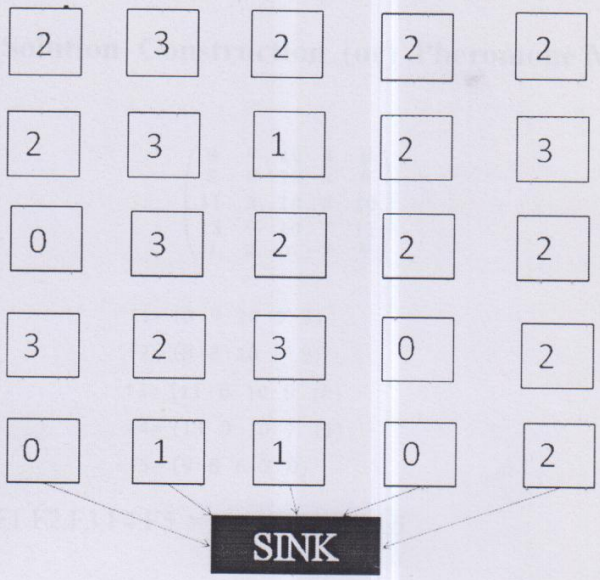
The algorithm for the classifier is then applied to each instance of the test set for each of the 10 iterations. The accuracy for each iteration is calculated which is then averaged out to get the classifiers accuracy.

In case of 10-fold cross validation, the accuracies obtained from all runs are averaged to get the average accuracy of the classifier. The standard deviation is also calculated to get the range of acceptable accuracy and to see if any of the accuracies are too high or too low [10].

Case Study

(1) Graph Construction





- $F_{1,1} - F_{2,3} - F_{3,2} - F_{4,5} - F_{5,1}$
  2-1-3-2-0 = 8
- $F_{1,1} - F_{2,1} - F_{3,1} - F_{4,2} - F_{5,5}$
  2-2-0-2-2 = 8
- $F_{1,1} - F_{2,2} - F_{3,4} - F_{4,1} - F_{5,2}$
  2-3-2-3-1 = 11
- $F_{1,1} - F_{2,5} - F_{3,2} - F_{4,1} - F_{5,5}$
  2-3-3-3-2 = 13
- $F_{1,1} - F_{2,4} - F_{3,5} - F_{4,2} - F_{5,3}$
  2-2-2-2-1 =9

ccccccc

(2) Solution Construction (or) Pheromone Matrix

$$\begin{pmatrix} 8 & 9 & 10 & 9 & 8 \\ 8 & 8 & 10 & 8 & 8 \\ 11 & 8 & 10 & 8 & 10 \\ 13 & 9 & 10 & 7 & 11 \\ 9 & 8 & 6 & 7 & 9 \end{pmatrix}$$

F1= (8 9 10 9 8)

F2= (8 8 10 8 8)

F3= (11 8 10 8 10)

F4= (13 9 10 7 11)

F5= (9 8 6 7 9)

F1,F2,F3,F4,F5 = feature 1,2,3,4,5

(3)Pheromone trails and heuristic information

- F-score heuristic information ,

$$\eta_i = \frac{\sum_{c=1}^{v}\left(\bar{x}^{(c)} - \bar{x}_i\right)^2}{\sum_{c=1}^{v}\{\frac{1}{N^{(c)}-1}\sum_{j=1}^{N_i^{(c)}}\left(x_{i,j}{}^{(c)} - \bar{x}_i{}^{(c)}\right)^2\}}$$

where , η_i =heuristic value

v =number of categories of target variable

$c \in \{1,2,3,\ldots,v\}$

c =categorical value

$j \in \{1,2,3,\ldots,N_f\}$

N_f = number of features

$\bar{x}^{(c)}$= mean of the feature with categorical value c

$\bar{x}_i$ = mean of the i^{th} feature

$N^{(c)}$ = number of samples of features with categorical value c

$N_i^{(c)}$ = number of samples of i^{th} feature with categorical value c

$j \in \{1,2,3,\ldots\ldots N_i^{(c)}\}$

$x_{i,j}^{(c)}$ = the j^{th} training sample for the i^{th} feature with categorical value c

$\bar{x}_i^{(c)}$ = mean of the i^{th} feature with categorical value c

$$\eta_1 = \frac{\sum_{c=1}^{v}\left(\bar{x}^{(c)} - \bar{x}_1\right)^2}{\sum_{c=1}^{5}\left\{\frac{1}{N^{(c)}-1}\sum_{i=1}^{N_1^{(c)}}\left(x_{1,j}^{(c)} - \bar{x}_1^{(c)}\right)^2\right\}}$$

$$\eta_1 = \frac{\{(10-9)^2 + (8-8)^2 + (9-9)^2 + (10-8)^2 + (8-9)^2\}}{\left\{\begin{array}{l}\left[\frac{1}{8-1}\{(8-9)^2 + (9-9)^2 + (10-9)^2 + (9-9)^2 + (8-9)^2\}\right] + \\ \left[\frac{1}{9-1}\{(8-9)^2 + (9-9)^2 + (10-9)^2 + (9-9)^2 + (8-9)^2\}\right] + \\ \left[\frac{1}{10-1}\{(8-9)^2 + (9-9)^2 + (10-9)^2 + (9-9)^2 + (8-9)^2\}\right] + \\ \left[\frac{1}{9}\{(8-9)^2 + (9-9)^2 + (10-9)^2 + (9-9)^2 + (8-9)^2\}\right] + \\ \left[\frac{1}{8-1}\{(8-9)^2 + (9-9)^2 + (10-9)^2 + (9-9)^2 + (8-9)^2\}\right]\end{array}\right\}}$$

$$\eta_1 = \frac{(1+0+0+4+1)}{\left\{\begin{array}{c}\left[\frac{1}{7}(1+0+1+0+1)\right] + \left[\frac{1}{8}(1+0+1+0+1)\right] + \\ \left[\frac{1}{9}(1+0+1+0+1)\right] + \left[\frac{1}{8}(1+0+1+0+1)\right] + \\ \left[\frac{1}{7}(1+0+1+0+1)\right]\end{array}\right\}}$$

$$\eta_1 = \frac{6}{\{0.42 + 0.37 + 0.33 + 0.37 + 0.42\}}$$

$$\eta_1 = \frac{6}{1.91}$$

$$\eta_1 = 3.14$$

CHAPTER 4
SYSTEM DESIGN AND IMPLEMENTATION

This chapter describes the flow of the system and form design and details explanation of the system.

4.1 System Design

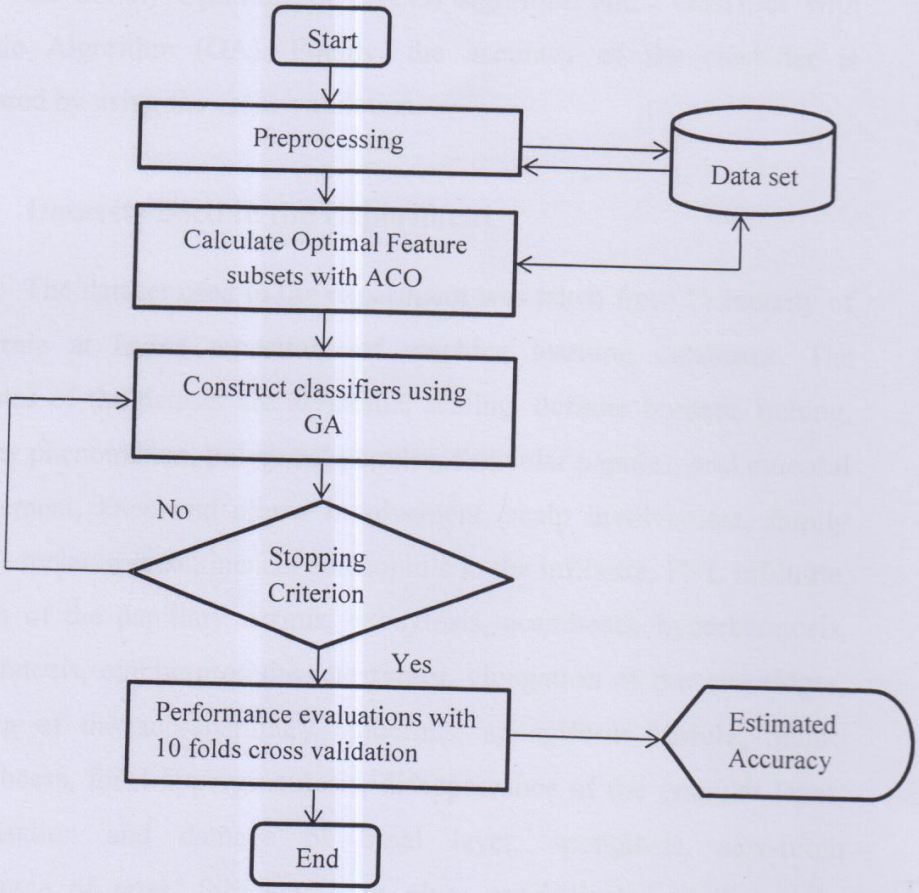


Figure (4.1) System Flow Diagram

System design is shown in Figure (4.1). First, this system finds the missing value using the mean-value in the data preprocessing step. Second, training set and test set is divided into 10 sets. Training set is used to build the classifier and test set is used to test the unknown dataset. Third, this system finds the good subset feature with filter approach by using Ant Colony Optimization (ACO) algorithm and , classifies with Genetic Algorithm (GA). Finally, the accuracy of the classifier is evaluated by using the cross-validation.

4.2 Datasets Used in the Experiment

The dataset used in the experiment was taken from University of California at Irvine repository of machine learning databases. The attributes of the dataset are erythema, scaling, definite borders, itching, koebner phenomenon, polygonal papules, follicular papules, oral mucosal involvement, knee and elbow involvement, scalp involvement, family history, melanin incontinence, eosinophils in the infiltrate, PNL infiltrate, fibrosis of the papillary dermis, exocytosis, acanthosis, hyperkeratosis, parakeratosis, clubbing of the rete ridges, elongation of the rete ridges, thinning of the suprapapillary epidermis, spongiform pustule, munro microabcess, focal hypergranulosis, disappearance of the granular layer, vacuolisation and damage of basal layer, spongiosis, saw-tooth appearance of retes, follicular horn plug, perifollicular parakeratosis, inflammatory monoluclear infiltrate, band-like infiltrate and Age.

In the dataset constructed for this domain, the family history feature has the value 1 if any of these diseases has been observed in the family and 0 otherwise. The age feature simply represents the age of the patient. Every other feature (clinical and histopathological) was given a

degree in the range of 0 to 3. Here, 0 indicates that the feature was not present; 3 indicate the largest amount possible, and 1, 2 indicate the relative intermediate values [3].

4.3 System Implementation

The system has four components: View Data, Preprocessing, Feature Selection and Classification.

4.3.1 View Dataset

If user selects "Whole Data" from "View Data" menu, a form appears as shown in Figure 4.2. In this form, user can view data by clicking "ViewWholeData" button. After ViewWholeData, the system displays the user requirement dataset into the grid view. And then, user can click "Next" button to go to the next step of the system, data preprocessing step.

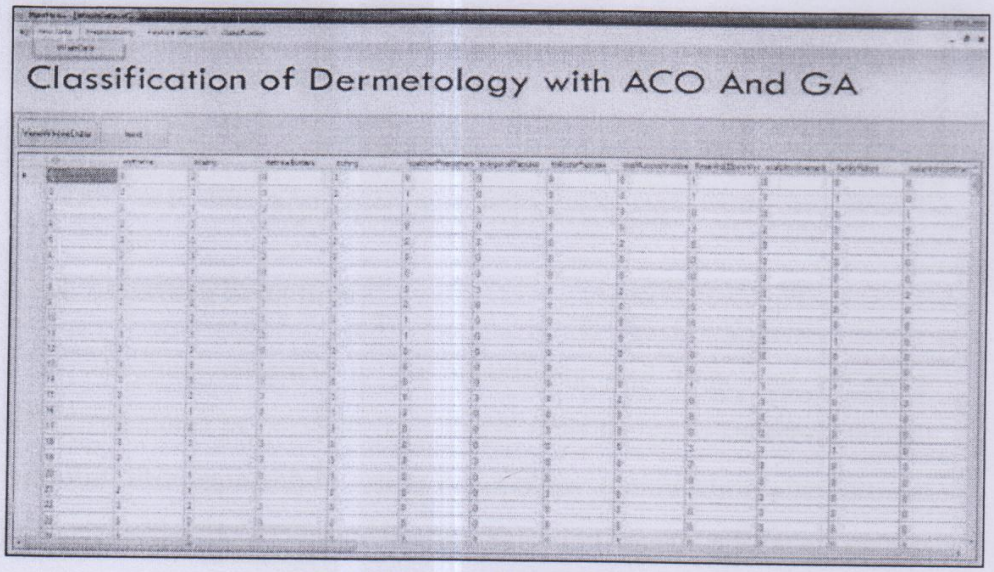


Figure (4.2) View Dataset Form

4.3.2 Data Preprocessing

If user selects "Preprocess Data" from "Preprocessing" menu, a form appears as shown in Figure 4.3. In this form, user clicks "PreprocessData" button to preprocess the dataset. In this thesis, dermatology datasets have missing values. For missing value, this thesis uses the attribute mean for all samples belonging to the same class as the given tuple. User can also click "Previous" button (to the back of the system's step) and "Next" button (to the next step).

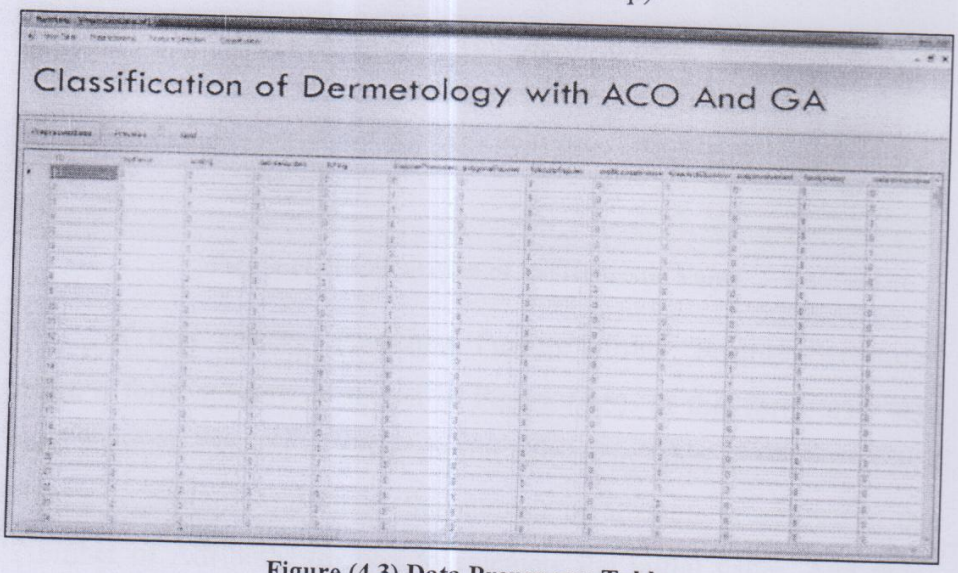


Figure (4.3) Data Preprocess Table

4.3.3 Feature Selection

Feature selection is the identifying and removing as much relevant features as possible. This thesis finds the good subset of feature with the filter approach by using Ant Colony optimization algorithm. If user selects the "ACO" from "Feature Selection" menu, a form appears as shown in Figure (4.4). In this form, user clicks "Calculate Heuristic Value" button to calculate the initial heuristic value of the features. The

system displays the heuristic value and relevant data into the grid view. User can also click "Previous" button (to the back of the system's step) and "Next" button (to the next step).

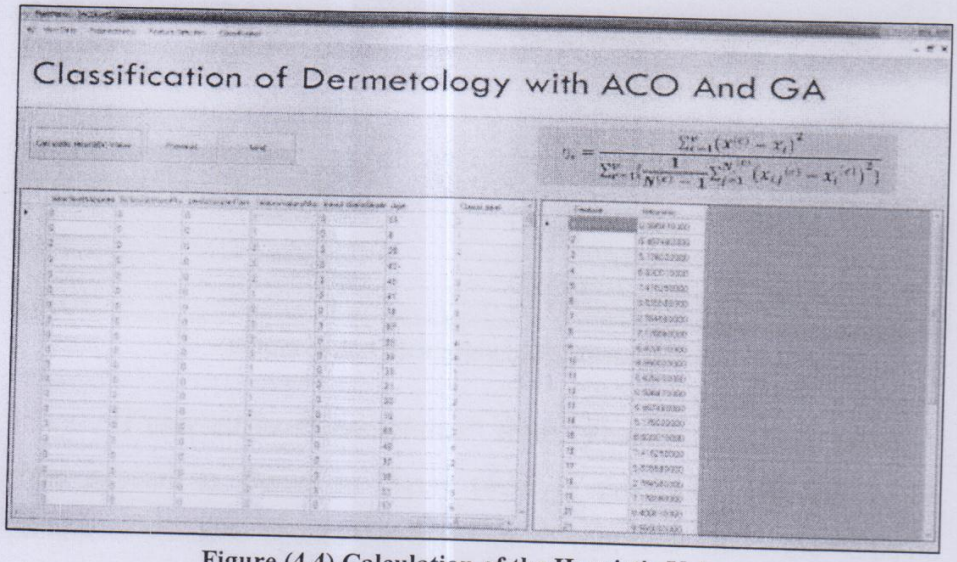


Figure (4.4) Calculation of the Heuristic Value

If user selects "Particular Feature" from "Feature Selection" menu, a form appears as shown in Figure (4.5). In this form, user sets good parameter values to calculate the particular value according to Ant Colony algorithm. And then, the number of attribute is entered in the "Number of Attribute" textbox to calculate the initial value. And then, user clicks "Calculate Particular value" button to calculate particular value. The system displays the particular value and relevant data into the grid view. User can also click "Previous" button (to the back of the system's step) and "Next" button (to the next step).

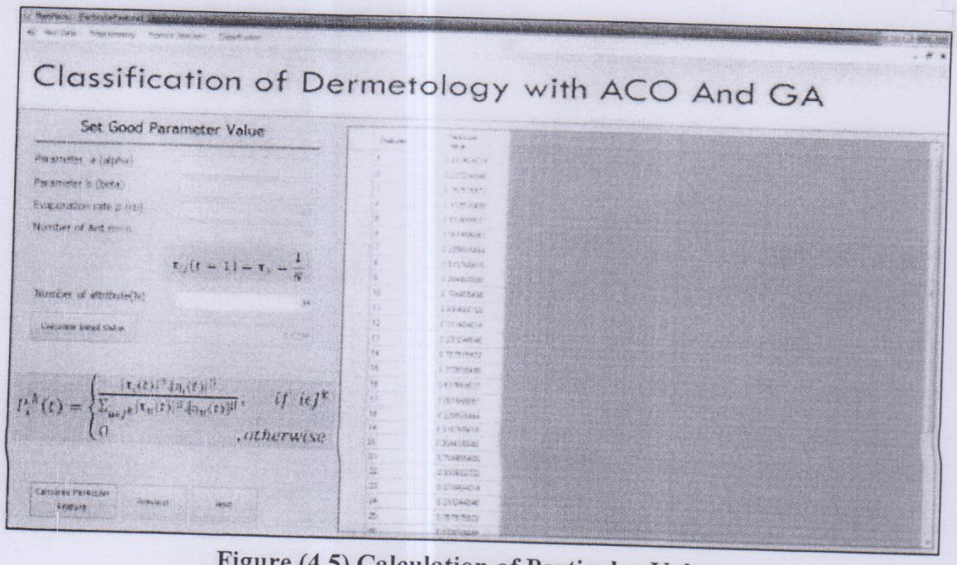


Figure (4.5) Calculation of Particular Value

If user clicks "UpdateFilter" from "Feature Selection" menu, a form appears as shown in Figure (4.5). In this form, user enters the parameter value to update the tour value. And, user clicks the "Update Tour Value" button to update tour value. Then, user chooses the highest tour value and clicks "Update Feature" button to update features. And then the threshold value is set to filter data and "Filter data" button to filter features. But, the numbers of selected features are varied according to the value of threshold. The system displays the updated tour value and updated feature values into the grid view.

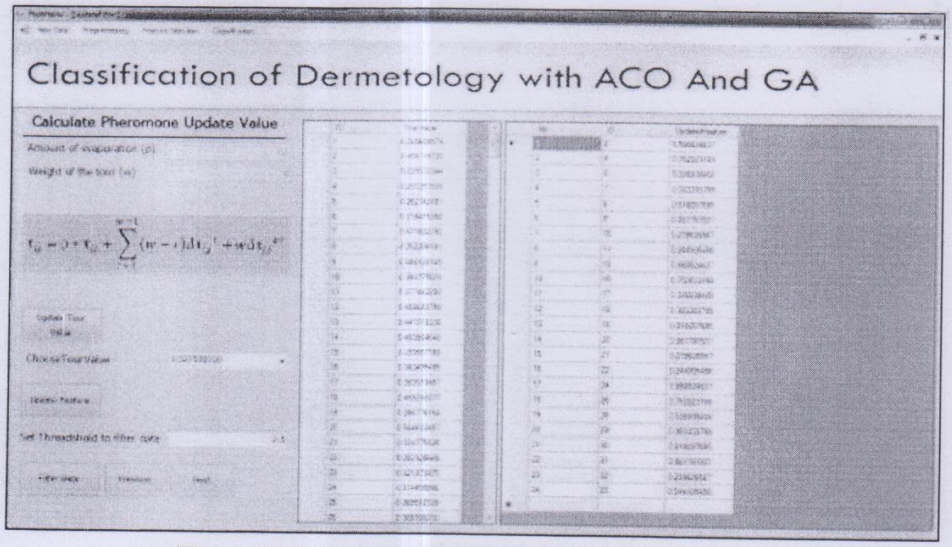


Figure (4.6) Update Tour value and Feature values

4.3.4 Classification

If user clicks "GAShowData" from "Classification" menu, a form appears as shown in Figure (4.6). User clicks "View Data" button to view data getting from ant colony algorithm. And then, user clicks "Transform Data" button to transform data value. The system displays the requirement dataset for genetic algorithm and updated feature values into the grid view. User can also click "Previous" button (to the back of the system's step) and "Next" button (to the next step).

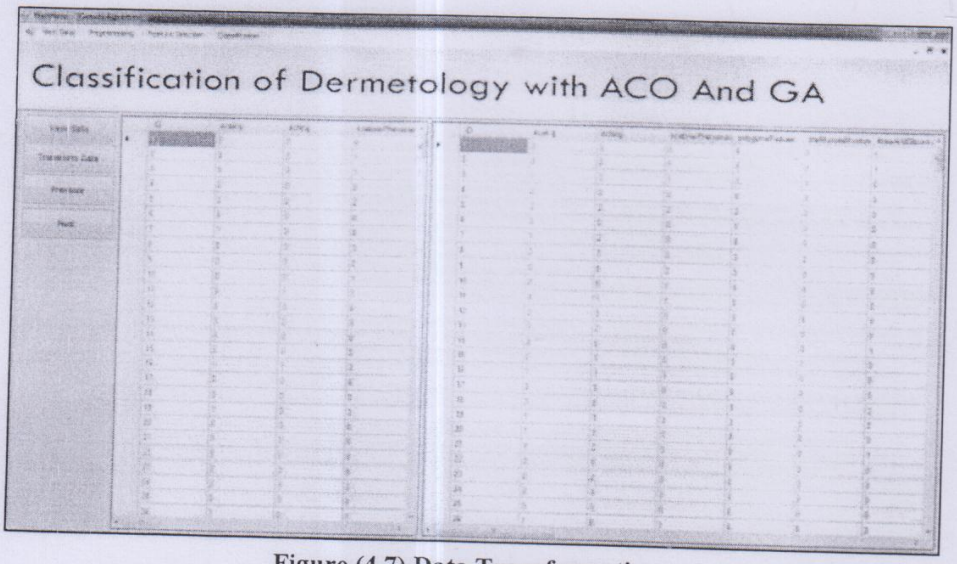


Figure (4.7) Data Transformation

If user clicks "Genetic Algorithm" from "Classification" menu as shown in Figure (4.7). If user clicks "CalculateGA" button, genetic algorithm calculates population size, generation, mutation rate and crossover rate by showing values to the corresponding textbox. If user clicks "Generate Rule" button , algorithm generates rules according to the records. The system displays rules into the label. User can also click "Previous" button (to the back of the system's step) and "Next" button (to the next step).

If user clicks "AccuracyRate" from "Classification" menu, a form appears as shown in Figure (4.9). The user must choose a "Rule" to classify data. Then, the system will generate the results of accuracy rate by using genetic algorithm (GA) and 10 fold cross validation method.

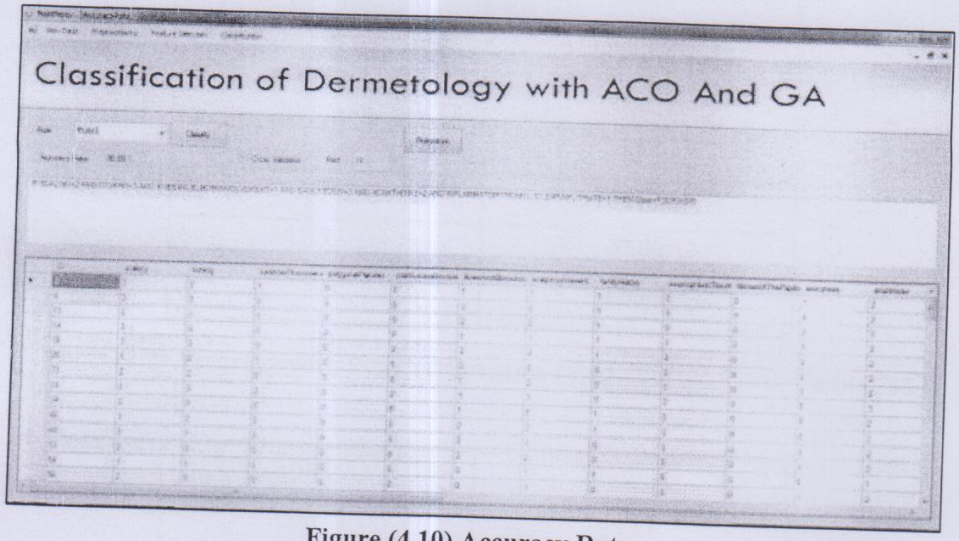


Figure (4.10) Accuracy Rate

10-Fold Cross-Validation	Accuracy
2	90.89
3	82.75
4	97.22
5	92.86
6	94.23
7	80.0
8	90.89
9	92.79
10	97.22

Table (4.1) Accuracy with Selected Features using 10-Fold Cross-Validation

CHAPTER 5
CONCLUSION, LIMITATION AND FURTHER EXTENSION

5.1 Conclusion

Machine learning has been applied to a variety of medical domains in order to improve medical decision-making. Machine learning provides methods, techniques and tools in a variety of medical domains. The feature selection process can improve the classification accuracy of the system. This thesis combines Ant Colony Optimization algorithm and Genetic Algorithm for evolving optimal subsets of discriminatory features. This thesis is to predict the accuracy of classification on dermatology dataset and to be effective in reducing dimensionality, removing irrelevant data and improving result. This thesis can be used to classify unknown datasets using the selected features.

5.2 Limitation and Advantages

This thesis can classify only on the Dermatology dataset. Dataset is mainly used from University of California at Irvine Repository of machine learning databases. This thesis uses filter approach of feature selection with Ant Colony Optimization algorithm and Genetic Algorithm.

In this thesis, the filter approach of feature subset selection presents to reduce the number of features with the maximum performance. Ant Colony Optimization Algorithm-based method improved the results by reducing the number of features required for learning the classification rules. If there is no expert, user can classify using the selected features with the unknown datasets. So, this thesis can classify with the new data

whether the patient will become psoriasis or seboreic dermatitis or lichen planus or pityriasis rosea or cronic dermatitis or pityriasis rubra pilaris.

5.3 Further Extension

This thesis can be extended to classify other Disease based on this thesis. In future, Dermatology problem can be solved by using other optimization algorithm such as Dijkstra's algorithm.

REFERENCES

[1] A.L. Blum an P.Langely. "Selection of relevant features and examples in machine learning", Artificial Intelligence, Vol. 69, pp.245-271, 1977.

[2] B. Calvo, P. Larannaga, J.A. Lozana. "Feature subset selection from positive and unlabelled examples". Pattern Recognition Letters, Vol. 30, pp. 1027- 1036, 2009

[3] G. Demiroz, H. A. Govenir, and N. Ilter, "Learning Differential Diagnosis of Eryhemato-Squamous Diseases using Voting Feature Intervals", Aritificial Intelligence in Medicine

[4] H.Jawei, K.Micheline, "Data Mining Concepts and Techniques", Simon Fraser University

[5] M.Dash and H.Liu, "Feature Selection for Classification," An International Journal of Intelligent Data Analysis, vol. 1, no. 3, pp.131-156, 1997.

[6] M.Marco Dorigo and Thomas Stutzle, " Ant Colony Optimization", Cambridge, Massachusetts London, England

[7] M.Maryam Bahojb Imani, Tahereh Pourhabibi, Mohammad Reza Keyvanpour, and Reza Azmi ,International Journal of Machine Learning and Computing, Vol. 2, No. 3, June 2012 .

[8] M. Dorigo, A. Colorni and V. Maniezzo, "The Ant System: optimization by a colony of cooperating agents," IEEE Transactions on Systems, Man, and Cybernetics-Part B, vol. 26, no. 1, pp. 29-41, 1996.

[9] M. Dorigo and G. Di Caro. The ant colony optimization meta-heuristic. In D. Corne, M. Dorigo and F. Glover, editors, new ideas in optimization, pages 11-32. McGrawHill, London, UK, 1999

[10] M. Anthony and S. B. Holden. Cross-validation for binary classification by real-valued functions: Theoretical analysis. In Proceedings of the International Conference on Computational Learning Theory, pages 218–229, 1998.

[11] S.Cateni, V.Colla, M.Vannucci, "Variable Selection through Genetic Algorithms for classification purpose", IASTED International Conference on Artificial Intelligence and Applications, 2010, Innsbruck, Austria, 15-17 February 2010.

[12] S.Spears William, The Role of Mutation and Recombination in Evolutionary Algorithms –dissertation for Doctor of Philosophy at George Mason University

[13] S.Cateni, V. Colla, M.Vannucci, "General Purpose Input Variable Extraction: a genetic algorithm based procedure GIVE A GAP", Proc of the 9th International Conference on Intelligent Systems Design and Applications, ISDA '09, November 30- december 2, 2009, Pisa, Italy

[14] Genetic Programming and Evolvable Machines, http://www.kluweronline.com/issn/1389-2576/contents