

CLASSIFICATION USING GRANULATION

*Dissertation submitted to the Jawaharlal Nehru University in partial fulfillment
of the requirement for the award of the degree of*

MASTER OF TECHNOLOGY
IN
COMPUTER SCIENCE AND TECHNOLOGY

BY

Surender Kumar Verma



SCHOOL OF COMPUTER AND SYSTEM SCIENCES
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI-110067, INDIA

2009



जवाहरलाल नेहरु विश्वविद्यालय
SCHOOL OF COMPUTER & SYSTEMS SCIENCES
JAWHARLAL NEHRU UNIVERSITY
NEW DELHI-67

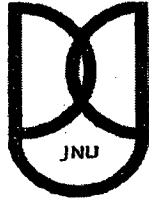
CERTIFICATE

This is to certify that this dissertation entitled “**Classification Using Granulation**” submitted by **Mr. Surender Kumar Verma**, to the **School of Computer and Systems Sciences**, Jawaharlal Nehru University, New Delhi, in partial fulfillment of the requirements for the award of degree of **MASTER OF TECHNOLOGY**, is a bonafide work carried out under my supervision.

The study pertains to this dissertation is original and has not been submitted, in part or in full, to any other University or Institution for the award of any other degree.

Prof. Sonajharia Minz
Dean,
SC&SS, JNU.

Prof. Sonajharia Minz
Supervisor
SC&SS, JNU.



जवाहरलाल नेहरु विश्वविद्यालय
SCHOOL OF COMPUTER & SYSTEMS SCIENCES
JAWHARLAL NEHRU UNIVERSITY
NEW DELHI-67

DECLARATION

This is to certify that the dissertation titled “**Classification Using Granulation**”, being submitted to the **School of Computer & System Sciences, Jawaharlal Nehru University, New Delhi**, in partial fulfillment of the requirements for the award of **Master of Technology in Computer Science & Technology** is a bonafide work carried out by me. This research work has been carried out the under the supervision of **Prof. Sonajharia Minz**.

The study pertaining to this dissertation is original and has not been submitted, in part or in full, to any other University or Institution for the award of any other degree.

(Surender Kumar Verma)

M. Tech, SC & SS, JNU,

New Delhi - 110067

Acknowledgments

First and foremost, with great pleasure I would like to express my heartfelt gratitude to Prof. Sonajharia Minz. Who undertook to act as my supervisor in spite of her busy schedule in academic and professional commitments. She has given me valuable guidance, ever-ready support, and steady encouragement throughout my Course. Under her supervision I have found great aid to my knowledge, mostly when it comes over insightful discussion on the ideas regarding topic. She supported me patiently, and directed me with her immense experience and wisdom.

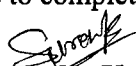
I owe my heartfelt gratitude to Prof. Parimala N, previous Dean, School of Computer & System Sciences, Jawaharlal Nehru University, New Delhi, for her kind and active cooperation during the course of study.

With a blend of gratitude, delight and gratification, I convey my indebtedness to all those who have directly or indirectly contributed to the successfully completion of my dissertation.

I would like to express my gratefulness to the entire faculty and staff of SC&SS for their cooperation during the course of study.

I also extend my thanks to all of my classmate and my lab mates especially Mr. Girish, Mr. Arun, and Mr. Ahmad, Mr. Kalicharan , Mr. Rajesh for their warmth, care and moral support.

Finally, I express my love and respect to my parents and all the family members, without their blessing and support it would not possible to complete this work.


(Surender Kumar Verma)

Abstract

Databases are rich with hidden information that can be used for making intelligent decision. Classification has always been the form of data analysis that can be used to extract models describing important data classes. Such analysis can help provide us the better understanding of the data at large. Many classification and prediction methods have been proposed by the researches in machine learning, pattern classification and statistics. Most algorithms are memory resident, typically assuming a small data size. Recent data mining research has build on such work, developing scalable classification and prediction techniques capable of handling large disk resident data.

In order to tackle such basis problem embedded in most of the data mining technique, the granular computing evolved. Although the term granular computing is recent but we have been studying the notions of granular computing in other forms like divide and conquer algorithms, structured programming etc.

The granular computing for classification have been explore here. The data input instead of directly feeding to the classifier to train the classifier, it is first granulated. Granules are formed using various granulation models. Granulation tree is one such popular model for constructing granules. The data tuples are then selected from these granules. The data can be randomly selected or some heuristic can be applied for the data selection.

The granule data is then applied on the classification algorithms to train the classifiers and the results of the experiments have been observed.

TABLE OF CONTENTS

ACKNOWLEDGMENT

ABSTRACT

LIST OF FIGURE

LIST OF TABLE

1. INTRODUCTION.....	1
1.1 Classification.....	2.
1.2 Granular Computing.....	2.
1.3 Problem Description.....	3
1.4 Dissertation Organization.....	3
2. CLASSIFICATION.....	5
2.1.1 Issues Regarding Classification.....	6
2.1.2 Various classification algorithms.....	7
2.1.3 Evaluating the Accuracy of Classifier.....	17
3. GRANULAR COMPUTING	
3.1 Basic components of Granular computing.....	20
3.1.1 Granules.....	21
3.1.2 Granulated View and Levels.....	21
3.1.3 Hierarchies	21
3.1.4 Models of granulation.....	21
3.2 Information processing pyramid.....	21
3.3 Encoding decoding mechanism.....	22
3.4 Relationship among granules,.....	23
3.4.1 Refinement and coarsening	22
3.4.2 Granulation as partition and covering.....	23
3.4.3 Partition.....	23
3.4.4 Covering.....	24

3.4.5	Partial ordering.....	24
3.4.6	Similarity relationship.....	24
3.5	Models of granulation.....	25
3.5.1	partition model.....	25
3.5.2	Zadeh's formulation framework.....	26
3.5.3	Pawlak's rough sets.....	27
3.5.4	Set theoretic models of GrC.....	28
3.5.5	Neighborhood System.....	29
3.5.6	Granulation tree.....	30
3.5.7	Factors affecting the process of granulation.....	32
3.5.8	Advantage of granular computing.....	32
4.	ALGORITHMS FOR GRANULAR COMPUTING	34
4.1	Proposed Framework.....	34
4.2	Granulation.....	35
4.3	Granule Representation.....	36
4.4	Implementation Details.....	37
5.	EXPERIMENT AND RESULTS	
5.1	System Description.....	39
5.2	Dataset description.....	39
5.3	Evaluation parameters.....	40
5.4	Description of Experiment	41
5.4.1	Results of classification without granulation.....	42
5.4.2	Granulation results.....	43
5.4.3	Results of classification using granulation.....	44
5.4.4	Comparison.....	46
5.4.5	Analysis.....	48
6.	CONCLUSION AND FUTURE WORK	
Future Work.....		50

REFERENCES	51
-------------------------	-----------

LIST OF FIGURE

Figure2.1 Decision Tree.....	7
Figure2.2 Artificial Neural Network.....	15
Figure3.1 Information processing pyramid.....	22
Figure3.2 Granulation tree.....	31
Figure4.1 Framework of Classification using Granulation.....	35
Figure4.2 Granulation.....	36
Fig4.3 Granule Representation.....	37

LIST OF TABLES

Table5.1: Data Description.....	40
Table5.2: Confusion Matrix.....	40
Table5.3 Accuracy without granulation.....	42
Table1.4 Precision without Granulation.....	43
Table5.5 Granulation Results	43
Table5.6 Accuracy with granulation.....	44
Table5.7 Precision with granulation.....	45
Table5.8 Accuracy Comparison.....	46
Table5.9 Precision Comparison.....	47

CHAPTER 1

INTRODUCTION

1.1 Classification

In real life we have to take certain decision that are based on the previous decisions. We keep the record of these decisions for the future reference so that we can make the better decisions about the categories of various real world entities. This phenomenon of the labeling the entities on the basis of prior decision is known as classification.

Databases contain information which can be used for making intelligent decisions. Classification and prediction are two most used forms of data analysis that can be used to make the important decision about the data and for prediction of data. The classification predict the possible class in which the given data may lie in. classification by decision tree induction and Bayesian classifiers are used in this dissertation for the classification purpose. Other classification algorithms like rule based classifiers, classification by artificial neural networks, support vector machine; k-nearest neighbor classifiers are also explained with appropriate details [1, 2].

1.2 Granular Computing

Granular computing is an approach of representing and processing basic chunks of information – information granules. Information granules, are collections of entities, usually originating at the numeric level, that are put together due to their functional adjacency, similar and indistinguishable characteristics [1, 3].

The essential factors that drive all pursuits of information granulation include a need to split the problem into a sequence of more manageable and smaller sub tasks. Here granulation serves as an efficient tool to granulize the problem. The primary intent is to reduce an overall computing effort and provide with a better insight into the problem rather than get buried in all unnecessary details about the problem. Here, granulation serves as a mechanism that reduces an entire burden of getting rid of unnecessary information into the dataset and helps in better understanding the problem. By changing

the “size” of the information granules we can hide or reveal a certain amount of details during a certain design phase.

1.3 Problem Description

The main concern of this dissertation is to design a model of classification using granulation approach. At present time there are many popular classification algorithms which are used in data mining. Most of these algorithms have the issue of scalability that is problem in handling large amount of data. Here comes the granular computing for the classification. The original dataset is first decomposed in to a set of clumps known as granules. These granules are then subjected to various data selection criteria. In experiment part of this dissertation random sampling from the granules is done. Then this granulated data is applied on various classification algorithms. The results are then analyzed against the ordinary classifiers. It has been noted that the granular computing reduced the data set and hence the computation time up to and considerable amount.

1.4 Dissertation organization

The dissertation is divided into five chapters. The chapter1 introduced the dissertation, granular computing and a detailed description of classification and various models and algorithms of classification.

Chapter2 gives a detailed description about the granular computing, why the granular computing should be approached, various models of granulation, how the different granules interact with each other (relationship between granules).

Chapter3 provides the details about the various algorithms that are used throughout this dissertation. The algorithm includes, algorithm to granularize the database, algorithm for construction of granulation tree, algorithm for selection of data tuples from various granules, algorithm to construct decision tree, algorithm to calculate

the significance of the attributes so that these can be applied for construction of optimum decision tree.

Chapter4 provides the details of result of experiment. The experiment is performed with selecting one, two and three tuples per granule. The granulated data is then applied on the classification algorithms. The various classification algorithms chosen are: Classification by decision tree induction, Bayesian classifiers, Classification by Backpropagation (by neural networks). The results of the experiments are shown in different tables.

Chapter 5 is about the conclusion and the scope of granular computing in other soft computing techniques.

CHAPTER2.

CLASSIFICATION

The task of classification involves assignment of decision values to the data objects based on the classifier learnt from the labeled training data objects using a suitable learning technique.

2.1 Issues Regarding Classification

a. Data Cleaning

Data cleaning is a process of removing the noise and outliers from the data based used for the classification. It includes treatment of missing values, reducing the noise and removing the outliers. Although most of the known classification algorithms have some mechanism to deal with all the above said problems, the data cleaning techniques may be applied on the data base [1, 2].

b. Relevance Analysis

All the information given in the database may not be relevant for the process of classification. Many of the attributes in the database may not help in the process of training a classifier. Correlation analysis can be used to identify whether two given attributes are statically related. A database may contain irrelevant attributes. For example employee_id can not help in the classification process in any way. Attribute subset selection can be used to find reduced set of attributes.

c. Data transformation and reduction

There are various data transformation techniques like normalization. Normalization involves scaling all values for a given attribute so that they fall within a small specified range such as 0.0 to 1.0. Data can be transformed by generalizing it to higher level concept. This is useful for continuous valued attributes. Discretization is also a form for data transformation. This is also useful in case of continuous valued attributes. For example the income attribute may be discretized to low, medium and high values.

2.1.1 Various Classification Algorithms

A. Classification by Decision Tree Induction

It is learning of decision tree from the class labeled training tuples. A decision tree is a flow chart like tree structure where search internal node (non leaf node) denoted a test on an attribute, each branch represents an outcome of the test and each leaf node (or terminal Node) holds class label. The topmost node in the tree knows as root of the tree. A typical decision tree is shown below:

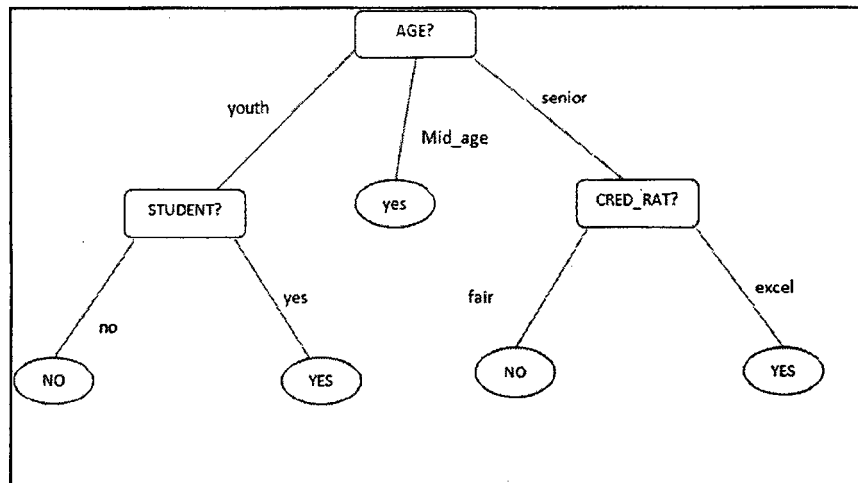


Figure 1.1 Decision Tree

Decision tree Induction

The various decision tree algorithms used for the classification purpose are: ID3, C4.5 and CART (classification and regression tree). The algorithm for decision tree induction is given in the algorithms section. Here is the detailed description of the algorithm.

The input to the algorithms is, Dataset, D , attribute list and the attribute selection methods. D is set of tuples each with a class label. Attribute list is the list of attributes

describing the set of attributes constituting the tuples of the dataset. Attribute selection measures the criteria on the basis of which the best attribute for decomposition of the data set is selected among the set of attributes.

The tree node starts with a single node, N, representing the training tuples in the data set D. If tuples in the D are of same class, then node N becomes a leaf node and class name is assigned as its label. Otherwise, the algorithm calls Attribute Selection methods to determine the best attribute for splitting the D. The splitting criteria tell which branches to grow from node N with respect to the outcome of the chosen test. The node N is labeled with the splitting criterion, which is a test at the node. The tuples in D are partitioned accordingly.

For an attribute A being a splitting attribute, there can be three possible scenarios based on the type of values the attribute A has:

- **A is discrete:** Values attribute: the outcome of the test at node N correspond directly to the known values of A. A branch is created corresponding to each value of the attribute. The partition D_j of D is the subset of class labeled tuples in D having value a_j of A.
- **A is continuous valued attribute:** A split point is created based on the continuous value of the attributes. This split point may be the mean or median of all the values of the attributes. The splitting node has two outcome at this point, one corresponding to $A \leq \text{split_point}$ and one $A \geq \text{split_point}$.
- **A is discrete valued attribute and a binary tree must be produced**

Following are the stopping criteria of the algorithm

- All the tuples in the partition D belong to the same class.
- There is no remaining attribute on which the tuples may be further partitioned. In this the majority voting is employed. It implies converting node N into a leaf and labeling it with the most common class D.
- There is no tuples in a partition. In this case a leaf is created with the majority class in D.

The complexity of the decision tree algorithms is $O(n \times |D| \times \log(|D|))$. Here n , number of attributes in the tuples, $|D|$ is the no. of tuples in the training database.

The various Attribute Selection Measures Are

i. Information Gain

The attribute with highest information gain minimizes the information needed to classify the tuples in the resulting partition and reflects the least randomness or "impurity" in these partitions. The expected information needed to classify a tuple in D is given by

$$Info(D) = - \sum_{i=1}^m p_i \log_2(p_i)$$

Here p_i is the probability that an arbitrary tuple in D belongs to class C_i and is estimated by: $|C_{i,D}| / |D|$. The $info(D)$ is also known as entropy or randomness of D .

Now suppose we have to partition the tuples in D on some attribute A having V discrete values, $\{a_1, a_2, \dots, a_v\}$. This attribute is used to partition the D into v subsets or partitions, $\{D_1, D_2, \dots, D_v\}$, where D_j contains those tuples in D that have outcome a_j of A .

The information of attribute, A on the partition D is given as:

$$Info_A(D) = \sum_{j=1}^v \frac{|D_j|}{|D|} \times Info(D_j)$$

Gain of *the* attribute A is given as:

$$Gain(A) = Info(D) - Info_A(D).$$

ii. Gain Ratio

There are the *situations* when an attribute has large number of values. For example the attribute `product_id`. If this attribute is subjected to the information gain criteria, then this will have highest information gain. When this attribute is used for splitting then there will be large no. of partition each having one tuple (being `product_id`

as the primary key of this database). But such kind of partition is never useful. In order to avoid such condition, the Gain ratio [2] as attribute selection measure is used.

SplitInfo represents the potential information generated by splitting the training data set, D, into v partitions, corresponding to the v outcomes of attest on attribute A.

SplitInfo of the attribute is calculated as:

$$\text{SplitInfo}_A(D) = \sum_{j=1}^v \frac{|D_j|}{|D|} \times \log_2 \left(\frac{|D_j|}{|D|} \right)$$

The gain ratio is given as:

$$\text{GainRatio}(A) = \frac{\text{Gain}(A)}{\text{SplitInfo}(A)}$$

The attribute with the maximum SplitInfo is selected as the attribute to split the data tuples in the corresponding partition.

iii. Gini Index

The *gini* Index is used in CART [2] decision tree. The *gini* index measures the impurity of a data partition. The *gini* index of a partition is given as:

$$\text{gini}(D) = 1 - \sum_{i=1}^m p_i^2,$$

Where p_i is the probability that a tuple in D belongs to class C_i and is estimated by, $|C_{i,D}| / |D|$.

The main feature of *gini* index is that it considers a binary split for each attribute. When considering a binary split we compute the weighed sum of the impurity of each resulting partition. For example, if a binary split on A partition D into D1 and D2, the *gini* index of D given that partitioning is

$$gini_A(D) = \frac{|D_1|}{|D|} gini(D_1) + \frac{|D_2|}{|D|} gini(D_2).$$

The reduction in impurity that would be incurred by a binary split on a discrete or continuous values attribute A is:

$$\Delta gini(A) = gini(D) - gini_A(D)$$

B. Bayesian Classification

Bayesian classifiers are statistical classifiers which can predict class membership probabilities, such as probability that a given tuple belongs to a particular class. Bayesian classifiers are based on the Bayes' theorem. The Bayes' theorem is described below:

Bayes' Theorem

Let X be data tuple. In the Bayesian terms, X is considered "evidence." X is described by the set of n attributes. Let H be some hypothesis, such that the data tuple X belongs to a specified class C. For classification problem we want to determine $P(H|X)$, the probability that the hypothesis H holds the "evidence", data tuple X. In the other word, we are looking for the probability that tuple X belong to the class C, given the description of the tuple in the form of attribute vector.

$P(H|X)$ is the posterior probability of H conditioned on X. $P(H)$ is the prior probability of H. The posterior probability, $P(H|X)$ is based on more information than the prior probability, $P(H)$, which is independent of X. Similarly $P(X|H)$ is the posterior probability of X conditioned on H. $P(X)$ is the probability of X. According to Bayes' theorem,

$$P(X|H) = \frac{P(X|H)P(H)}{P(X)}$$

This theorem is used in Naïve Bayesian Classification as:

Naïve Bayesian Classification

Let D be a training set of tuples and their associated class label. Each tuple is an n -dimensional attribute vector, $X=(x_1, x_2, \dots, x_n)$.

Suppose that there are m classes C_1, C_2, \dots, C_m . Given a tuple, X , the classifier will predict that X belong to the class having the highest posterior probability, conditioned on X . That is, the naïve Bayesian classifier predicts that tuple X belong to the class C_i if and only if

$$P(C_i|X) > P(C_j|X) \text{ for } i \leq j \leq m, j \neq i$$

Thus we maximize $P(C_i|X)$. The class C_i for which $P(C_i|X)$ is maximized is called the maximum posterior hypothesis.. By Bayes' theorem,

$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}$$

$$\text{Here } P(C_i|X) = \frac{|C_{i,D}|}{|D|}$$

Given set with many attributes, it would be extremely computationally expensive to compute $P(X|C_i)$. In order to reduce the computation the naïve assumption of class conditional independence is made. It presumes that the values of the attributes are conditionally independent of one another. Thus,

$$\begin{aligned} P(C_i|X) &= \prod_{k=1}^n P(x_k | C_i) \\ &= P(x_1|C_i) \times P(x_2|C_i) \times \dots \times P(x_n|C_i) \end{aligned}$$

Here x_k refers to the value of attribute A_k for tuple X .

- a. If A_k is categorical, then $P(x_k | C_i)$ is the number of tuples of class C_i in D having the value x_k .
- b. If A_k is continuous valued attribute then we need to have a Gaussian distribution with mean μ and standard deviation σ defined by:

$$g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

So that $P(x_k | C_i) = g(x_k, \mu_{c_i}, \sigma_{c_i})$.

C. Rule Based Classification

Rule based classification is a ruled based model represented as a set of IF-THEN rules. Rules are good way representing information or bits of knowledge. An IF-THEN rule as an expression of the form:

IF condition THEN conclusion.

For example a rule R,

R: IF age=youth AND student=yes THEN buy_computer = yes.

The IF part of rule is known as rule antecedent are precondition. The THEN part is the rule consequent. In the rule antecedent the condition on one or more attribute is defined. If the condition in a rule antecedent holds true for a given tuple, we say that the rule antecedent is satisfied and the rule covers the tuple.

A rule can be accessed by the two terms: coverage and accuracy. Let n_{covers} be the number of tuples covered by R; $n_{correct}$ be the number of tuples correctly classified by R; and $|D|$ be the number of tuples in D . We can define the coverage and accuracy of R as:

$$\text{Coverage}(R) = \frac{n_{covers}}{|D|}$$

$$\text{Accuracy (R)} = \frac{n_{\text{correct}}}{n_{\text{covers}}}$$

That is the rule coverage is the percentage of tuples that are covered by the rule. For rule accuracy, we look at the tuples that it covers and see what percentage of then the rule can correctly classify.

Rule Quality Measures

The obvious chose for the rule quality measure is accuracy. Sometime the accuracy measure is not sufficient. There are other rule quality measures like FOIL_gain and Likelihood Ratio. The details are not given here.

D. Classification By Artificial Neural Networks

A multilayer feed forward network iteratively learns a set of weights of prediction of the class labels of the tuples. It contains one input layer, one or more hidden layer and one output layer.

The input to the network corresponds to the attributes measured for each training tuple. The inputs are fed simultaneously to the input layer. These inputs are passed through the input layer and then weighed and fed simultaneously to the units of second layer that is to the first hidden layer. And the output of the first hidden layer is again assigned the weights and this is passed as an input to the second hidden layer of to the output layer in case there is only one hidden layer [2].

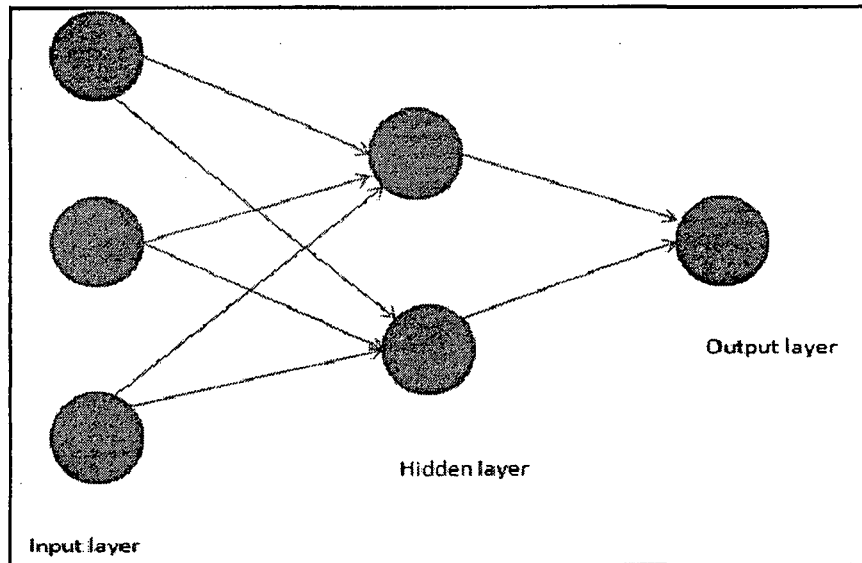


Figure2.2 Artificial Neural Networks

Network Topology

The neural network can be used for both classification and prediction. There is need to specify the number of nodes in the input layer, number of hidden layer and the number of units in each hidden layer and the number of units in the output layer before training the network. The input value may be normalized so as to fall between 0.0 and 1.0. The discrete values may be encoded so that each input unit per domain value. The continuous value may be discretized.

The best number of hidden layer is not restricted to any clear rule. The initial value of weights may affect the resulting accuracy of the network. Once a network is trained and it does not provide the expected accuracy the network is trained again with the different network topology and with the different set of weights.

Backpropagation

Backpropagation learns by iteratively processing of data set of training tuples, comparing the network's prediction for each tuple with the known target value. The target value here may the class label or a continuous value in case of prediction [1].

E. Support Vector Machine

Support Vector Machine is a method of classification for both linear and non linear data. The working principle of Support Vector Machine (SVM) is like; it uses a nonlinear mapping to transform the original data into higher dimensions. With this approach of transformation two classes can also be separated with the help of a hyper plane. Here the support vectors are the essential training tuples and the margins (defined by the margins).

The main feature of SVM is to find the hyper plane using these support vectors.

The essential features of SVM are:

1. The training time of SVM is extremely slow,
2. are highly accurate for classification
3. Capable to model complex nonlinear decision boundaries.
4. Are much less prone to over fitting than other methods of classification.

The Support Vector Machines are designed to handle for linearly separable and nonlinearly separable data. The both cases are considered here:

F. *K*-Nearest –Neighbor Classifier

Nearest neighbor classifiers are based on learning by analogy, that is, by comparing a given test tuple with training tuples that are similar to it. The training tuples are described by n attributes. Each tuple represents a point in an n -dimensional space. In this way all of the training tuples are stored in an n -dimensional pattern space. When given an unknown tuple, a *k*-nearest –neighbor classifier searched the pattern space for the k training tuples that are closet to the unknown tuple.

The closeness is defined in terms is a distance metric, such as Euclidean distance. The Euclidean distance between two points or tuples, say,

$$X_1 = (x_{11}, x_{12}, \dots, x_{1n})$$

And

$$X_2 = (x_{21}, x_{22}, \dots, x_{2n})$$

is given as:

$$\text{dist}(X_1, X_2) = \sqrt{\sum_{i=1}^n (x_{1i} - x_{2i})^2}$$

2.1.2 Evaluating the Accuracy of Classifier

Holdout, cross-validation and bootstrap are common techniques for assessing the accuracy based on randomly sampled partition of the given data. Use of such techniques to estimate accuracy increase the overall computation time.]

Cross-Validation:

In **k-fold cross-validation**, the initial data are randomly partitioned into k mutually exclusive subset D_1, D_2, \dots, D_k , each of approximately equal size. The training and testing is performed k times. In iteration i , partition D_i is reserved as the test set and the remaining partitions are collectively used to train the model; in second iteration.

Leave-one-out is a special case of k -fold cross-validation where k is set to the number of initial tuples. That is only one sample is left out at a time for the rest set. In Stratified cross validation, the folds are stratified so that the class distribution of the tuples in each fold is approximately the same as that in initial data. **10 fold Cross-validation** is recommended for estimating accuracy due to its relatively low bias and variance[1].

CHAPTER 3

GRANULAR COMPUTING

Granular computing is an approach of representing and processing basic chunks of information – information granules. Information granules are collections of entities, usually originating at the numeric levels that are put together due to their functional adjacency, similar and indistinguishable characteristics [4, 5].

The essential factors that drive all pursuits of information granulation include a need to split the problem into a sequence of more manageable and smaller sub tasks [3]. Here granulation serves as an efficient tool to granulate the problem. The primary intent is to reduce an overall computing effort and provide with a better insight into the problem rather than get buried in all unnecessary details about the problem. Here, granulation serves as a mechanism that reduces an entire burden of getting rid of unnecessary information into the dataset and helps in better understanding the problem. By changing the “size” of the information granules we can hide or reveal a certain amount of details during a certain design phase.

Although the granular computing has emerged recently, we have been studying the basic notion of granular computing in the name of divide and conquer, interval computing, quantization, artificial intelligence, structured programming, data compression, chunking, fuzzy and rough set theories, cluster computing, quotient space theory, belief functions, machine learning etc.

3.1 Basic Components of Granular Computing (GrC)

3.1.1 Granules

The granules are basically the modules of the problem in which the problem is divided. The consideration of granularity is motivated by the practical need for simplification, clarity, cost and tolerance of uncertainty. The computational complexity of problem solved with granulation is much less than the problem solved without granulation [5, 16, 7].

3.1.2 Granulated view and levels

A problem that is granulated can be further granular up to some higher level of granularity depending upon the size and time of computation. The multiple level of granulation covers the abstraction in terms of concreteness and amount of detail. Granules are formed with respect to a particular degree of granularity or detail. Granules in a level are defined and formed within a particular context and are related to granules in other levels [8].

3.1.3 Hierarchies

Granules in different levels are linked by the order relations and operations on granules. A granule in higher level can be decomposed into many granules in a lower level, and conversely many granules in a lower level can be combined into one granule in a higher level [9, 5].

3.1.4 Models of granulation

A problem can be divided into the granules with the help of various models of granulation. Some of models of granulation are relation based model, neighborhood system, covering based model etc [2, 8].

3.2 Information Processing Pyramid

In granular computing we operate on information granules. Information granules exhibit different levels of granularity. Depending upon the problem at hand, we usually group granules of similar “size” (that is granularity) together in a single layer. If more detailed (and computationally intensive) processing is required, smaller information granules are sought. Then these granules are arranged in another layer. In total, the arrangement of this nature gives rise to the information pyramid [5].

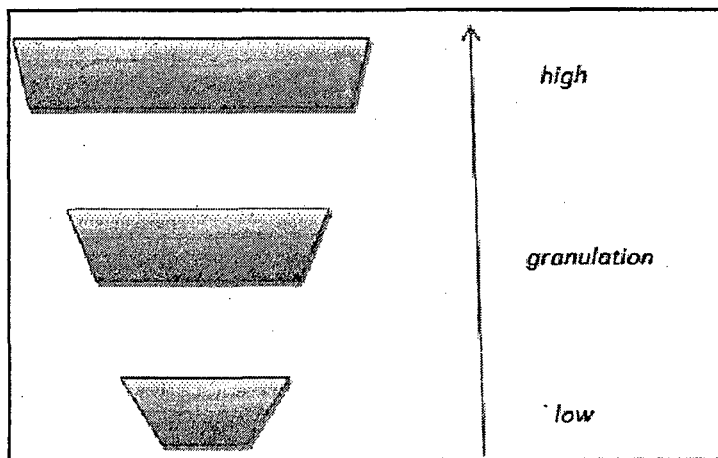


Figure 3.1 Information processing pyramid

- At the lowest level we are concerned with numeric processing. This is a domain completely overwhelmed by numeric models such as differential equations, regression models, neural networks, etc.
- At the intermediate level we encounter larger information granules (viz. those embracing more individual elements).
- The highest level can be solely devoted to symbol based processing and as such invokes well-known concepts of finite state machines, bond graphs etc.

3.3 Encoding Decoding Mechanism

When a problem is granulated in the various levels of hierarchy, the layers need to communicate with each other. They receive data from one layer as input, perform the computing on that data and return that data to some other layer. This process of communication is called as encoding/decoding mechanism. Encoder transforms the input into the form that is acceptable by the layer. The decoder converts

014.65
V59
cl

21

TH-17474



the information granule produced by given layer into the form that is acceptable by the destination layer [5].

The general formulations of the encoding – decoding problem can be thought as

$$\text{Dec}(\text{Enc}(X)) = X$$

3.4 Relationship Among Granules

3.4.1 Refinement and Coarsening

Granular computing involves structured human thinking. A high level granule represents a more abstract concept and a low level a more specific concept. The level of abstraction may be represented in terms of coarse and fine granules. A granule O_1 is a refinement of another granule O_2 , or equivalently O_2 is a coarsening of O_1 , denoted by $O_1 \preceq O_2$, or $O_2 \succeq O_1$ if every sub-granule or object of O_1 is contained in some sub-granules of O_2 . We call ' \preceq ' a fine relationship, and ' \succeq ' a coarse relationship. These relationships are also known as form and enclosure, i.e. O_1 forms O_2 and O_2 encloses O_1 .

3.4.2 Granulations as Partitions and Coverings

Partitions and coverings are two simple and commonly used granulations of a universe. They can be defined in the following way if a set-theoretical approach is Adopted [10]. A partition of a finite universe U is a collection of non-empty and pair wise disjoint subsets of U whose union is U . Each subset in a partition is also called a block or an equivalence class.

3.4.3 Partition For the partition,

$$\pi = \{X_i | 1 \leq i \leq m\}$$

The conditions of partition are:

(i) $X_i \neq \phi$

$$(ii) \quad X_i \cap X_j = \phi \quad \forall i \neq j$$

$$(iii) \quad \forall i \quad \cup \{X_i | 1 \leq i \leq m\}$$

3.4.4 Covering

For the covering,

$$\tau = \{X_i | 1 \leq i \leq m\}$$

The conditions of covering are:

$$(i) \quad X_i \neq \phi$$

$$(ii) \quad \forall i \quad \cup \{X_i | 1 \leq i \leq m\}$$

3.4.5 Partial Ordering

When comparing granules with fine or coarse relationships, every sub-granule within a finer granule is contained in the coarse granule. However, it may not be the case for all granules. In some cases, only some sub-granules of a fine granule are contained in the coarse granule. Therefore the fine and coarse relationship are not fully true but partially true. We call these relationships partial fine (p-fine) and partial coarse (p-coarse) relationships.

3.4.6 Similarity Relationship

Similarity is a key to forming an intra-relationship of a granule. It can also be used to measure closeness amongst granules. Various distance measures can be used to calculate how similar two granules are. Like in cluster analysis, similarity between two granules can be defined as an average distance between sub-granules. Similarities are usually normalized between 0 and 1. A similarity with a value 0 means that two granules are totally different with no overlap sub-granules. A similarity with

value 1 means they are the same granules with identical sub granules and structure. The similarity relationship is given as:

$$Sim(O_1, O_2) = \frac{1}{m \times n} \sum_{i=1, j=1}^{m, n} Sim(O_{1i}, O_{2j})$$

3.5 Models of Granulation

How the granulation is acquired from a given dataset? The answer of this question is various models of granulation. Each granulation model is explained in detail as given here:

3.5.1 Partition Model

This model is based on the definition of equivalence relations. Let U denote a finite and non-empty set called the universe. Suppose $E \subseteq U \times U$ denote an equivalence relation on U , where \times denotes the Cartesian product of sets. The essential properties of an equivalence relation are: reflexive, symmetric and transitive. The equivalence relation E partitions the U in various subsets of U denoted by $\pi_E = U/E$. The relation $x E_\pi y \Leftrightarrow x$ and y are in the same block of the partition, π [8, 11, 6].

One can define an order relation on the set of all partitions of U , or equivalently the set of all equivalence relations on U . A partition π_1 is a refinement of another partition, π_2 , or equivalently, π_2 is a coarsening of π_1 , denoted by $\pi_1 \preceq \pi_2$.

For the purpose of demonstration, an information table is a quadruple:

$$S = (U, A_t, \{V_a \mid a \in A_t\}, \{I_a \mid a \in A_t\})$$

U is a finite nonempty set of objects,

A_t is a finite nonempty set of attributes,

V_a is a nonempty set of values for $a \in A_t$,

$I_a : U \rightarrow V_a$ is an information function.

For a value $v \in V_a$, one obtains a granule with respect to an atomic formula:

$$m(a, v) = \{x \mid I_a(x) = v\}$$

It consists of all objects whose value on attribute a equals to v , and may be interpreted as a granule. Here all the partition of Universe are the equivalence classes of the relation "EQUAL TO"(the value of attribute). If the attribute $A = \{\text{Hair}\}$ is chosen, we can partition the universe into equivalence classes:

$$\{O_1, O_2, O_6, O_8\}, \{O_3\}, \{O_4, O_5, O_7\}$$

Finer Granules can be obtained by considering more than one attribute at time. e.g Height and Hair color gives

$$\{O_1, O_2, O_8\}, \{O_3\}, \{O_4, O_5, O_7\}, \{O_6\}.$$

3.5.2 Zadeh formulation Framework

Zadeh's formulation about the granular computing is based on fuzzy set theory. Fuzzy graphs and fuzzy if-else rules are used to represent the relationship between granules [12, 13].

Let X be a variable in the universe of values. $X \overset{isr}{R}$ represents a generalized constraint on the value of X , here R is constraining relationship, isr is a variable copula and r is a discrete variable whose value defines the way in which R constrains X . With these generalized constraints, a granule is defined by a set (fuzzy set) as:

$$G = \{X \mid X \overset{isr}{R}\}$$

The granule here may be labeled with the natural language words. The core concept of fuzzy logic is *one-to-one* correspondence between the granules and its

name is natural language. This is the essence of getting the granulation criteria from the natural concept.

Computation with words deals with fuzzy if-then else rules of the form

$$\text{if } X \text{ is } r_1 \text{ } A \text{ then } Y \text{ is } r_2 \text{ } B$$

Here r_1 and r_2 are different constraints. In general, the similar constraints are preferred. The fuzzy if-then else rules or fuzzy graphs are used to carry out the inference.

3.5.3 Pawlak's Rough Sets

In the terms of granular computing, a granule is considered as complete set of information rather than an individual entity of dataset defined arbitrarily. If some subset of information is described approximately then there is some loss in the process of granulation [5, 14].

Let $E \subseteq U \times U$ denotes an equivalence relation on universe U . The pair (U, E) is called as approximation space. E partitions U into disjoint equivalence classes. These classes may be considered as each individual granule of indistinguishable elements or objects. Two objects are indistinguishable if they have values with respect to the set of attributes.

An arbitrary set $X \subseteq U$ may not necessarily be union of some equivalence classes. This implies X may not be described using equivalence classes of E . In this case X may be characterized by upper and lower approximations as:

$$\overline{\text{apr}}(X) = \bigcup_{[x]_E \subseteq X} [x]_E$$

$$\underline{apr}(X) = \bigcup_{[x]_E \cap X \neq \emptyset} [x]_E$$

3.5.4 Set Theoretic Models of GrC

In Set theoretic model of granular computing, each granule represents a certain concepts such that each object of the granule represents the instance of the concept. The various set theoretic models are [15, 16]:

i. Power Algebras

For some *binary operation* O , the O^+ is defined as follows:

$$XO^+Y = \{xOy \mid x \in X, y \in Y\}$$

for any $X, Y \in U$.

In general, one may lift any operation f on elements of U to an operation f^+ on subsets of U , called the *power operation* of f . Suppose $f: U^2 \rightarrow U (n \geq 1)$ is an n -ary operation on U . The power operation f^+ is defined as:

$$f^+(X_0, X_1, \dots, X_n) = \{f(x_0, x_1, \dots, x_{n-1}) \mid x_i \in X_i \text{ for } i = 0, 1, \dots, n-1\}$$

For any $X_0, \dots, X_{n-1} \subseteq U$ s

The function f may impose some properties on its power function f^+ . For binary operation $F: U^2 \rightarrow U$ if f is commutative and associative, f^+ is commutative and associative. If f is *unary* operation on *involution* i.e $f(f(x)) = x$, then f^+ is also *involution*.

ii. Interval Number Algebra

An interval number $[a, a]$ with $a \leq a$ is the set of real numbers defined by [7, 11]:

$$[\underline{a}, \bar{a}] = \{x \mid \underline{a} \leq x \leq \bar{a}\}.$$

The arithmetic operation can be performed on the interval numbers by lifting arithmetic operation on real numbers.

$$\text{Let } A = [\underline{a}, \bar{a}] \text{ and } B = [\underline{b}, \bar{b}],$$

Be the two interval numbers. The set theoretic operation on these numbers can be defined as:

$$A + B = \{x + y \mid x \in A, y \in B\}.$$

$$= [\underline{a} + \underline{b}, \bar{a} + \bar{b}].$$

$$A - B = \{x - y \mid x \in A, y \in B\}.$$

$$= [\underline{a} - \bar{b}, \bar{a} - \underline{b}].$$

$$A \div B = \{x \div y \mid x \in A, y \in B\}.$$

$$= [\underline{a}, \bar{a}] \cdot [1/\bar{b}, 1/\underline{b}], \quad 0 \notin [\underline{b}, \bar{b}].$$

$$A \cdot B = \{x \cdot y \mid x \in A, y \in B\}$$

$$= [\min(\underline{a}\underline{b}, \underline{a}\bar{b}, \bar{a}\underline{b}, \bar{a}\bar{b}), \max(\underline{a}\underline{b}, \underline{a}\bar{b}, \bar{a}\underline{b}, \bar{a}\bar{b})].$$

3.5.5 Neighborhood System

A *crisp/fuzzy neighborhood system (NS/FNS)* to each object $p \in V$ (object space), we associate an (empty, finite or infinite) family of *crisp/fuzzy* [2] subsets of U (data space), called clumps. The mathematical system defined by these families is called *crisp/fuzzy neighborhood system* or simply *neighborhood system*, and these clumps associated to p are called *fundamental neighborhoods* of p . One can regard a fundamental neighborhood as a granule, so NS/FNS is locally a multiple level granulation [20, 21].

We will illustrate the idea by examples. Let $V = \{v_0, v_1, \dots, v_9\}$ and $COV = \{A_1, A_2, A_3, A_4\}$ a family of fuzzy sets of U that covers V . COV is a *fuzzy*

neighborhood system, in which each cover is a fuzzy neighborhood of any point in the cover.

$$\begin{aligned}
 FNS(v_0) &= \{A_1\}, & FNS(v_1) &= \{A_1, A_2\}, \\
 FNS(v_2) &= \{A_1, A_2\}, & FNS(v_3) &= \{A_3\}, \\
 FNS(v_4) &= \{A_2, A_3\}, & FNS(v_5) &= \{A_2, A_3\}, \\
 FNS(v_6) &= \{A_3\}, & FNS(v_7) &= \{A_3, A_4\}, \\
 FNS(v_8) &= \{A_3, A_4\}.
 \end{aligned}$$

The association,

$$p \rightarrow \{A_i\} \rightarrow \{name(i) = name(A_i)\} \text{ is a multiple valued representation of the universe.}$$

3.5.6 Granulation Tree

Tree is one of the best data structure to summarize the concept of hierarchy. It has good interpretability and well organized set of procedures for processing the data stored inside the tree. A granulation Tree is a framework for construction of granules from a dataset. The attributes of dataset are used to decompose the dataset. The root node is described by the universe of dataset. The concept of finer and coarse granules can be described with the help of granulation tree [12, 19].

The finer granules are approached when we traversal form root node to the leaf node of the tree. Every leaf node is the finest granule which can not be decomposed further. Similarly the coarse granules can be approached by traversing the tree from leaf node to root node. Typical granulation tree of Irish dataset is given here in Figure3.2.

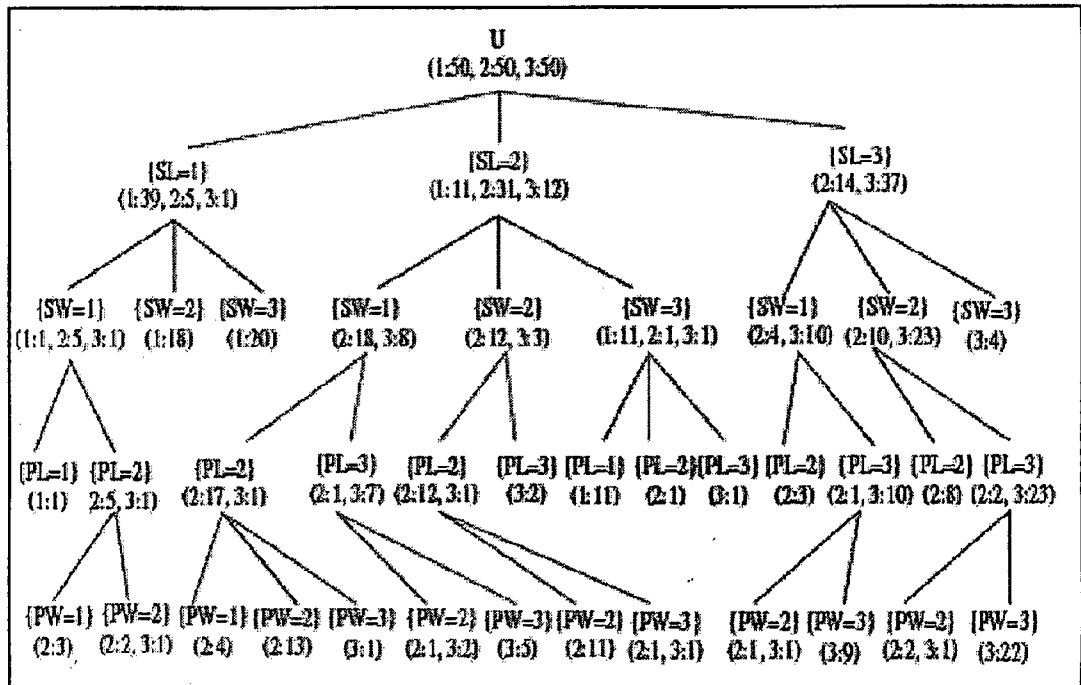


Figure3.2 Granulation Tree for iris dataset

Granulation Tree Construction

The root of granulation tree represents the universe of the data set. Set of attributes $A = \{A_1, A_2, \dots, A_n\}$ are used for decomposition of the data set in the root node. Two techniques can be employed here. First, attributes are selected at random for decomposition with any attribute selection approach and the selected attribute is used for the decomposition. Since no attribute selection criteria are employed here, the complexity of granulation tree construction can be reduced significantly. Disadvantage of this approach is there is no guarantee of getting good granules [19].

In second approach, some heuristic technique is employed for selection of attribute to be used for granulation. Such heuristic techniques can be, entropy of attribute, gain of attribute, gain ratio, gini index etc. Some attribute selection criteria based on roughest theory can also be applied. The attribute selection measure adds extra complexity on the granulation tree construction algorithm. Advantage of this approach is that good quality granules can be ensured.

3.6 Factors affecting the process of granulation

- (i) Granulation Criteria
- (ii) Representation of Granules
- (iii) Qualitative and quantitative characteristics of granules.
- (iv) Granulation algorithms and methods

3.7 Advantage of Granular computer

1. It helps to avoid confusion in pure algorithmic data processing while taking full advantage of the advances in algorithmic data processing.
2. It justifies the hyper computational nature of computing.
3. The size of dataset can be reduced to a great extent with the help of granulation process.
4. The problem definition can be modified according the convenience of model of granulation.

CHAPTER 4

**ALGORITHMS FOR CLASSIFICATION
USING GRANULATION**

The proposed framework for Classification using Granulation consists of two important components besides the optional components such as data preprocessing and visualization of extracted knowledge. These components are,

1. Granulation
2. Classifier extraction

4.1. Proposed Frame Work

In this dissertation process for granulation is employed on the training dataset. The granulation process is carried out with the help of granulation tree [18]. Since the outcome of granulation process is input to the next major component i.e training the classifier, the issue of granule representation needs to be addressed. For this the input specification of the classification algorithms need to be checked and if needed the encoding of granules or representation of granules need to be considered.

If the set of granules are used to train the classifier using classification algorithm then the resulting classifier would also be encoded during the granulation process. As the granulated data has been encoded, the output of the classifier may require being decoded in order to use the classifier for labeling the test data.

The test data may be applied to the classifier for classification and the result of the classifier may be observed before termination. A flow diagram of the proposed framework has been presented in Figure4.1.

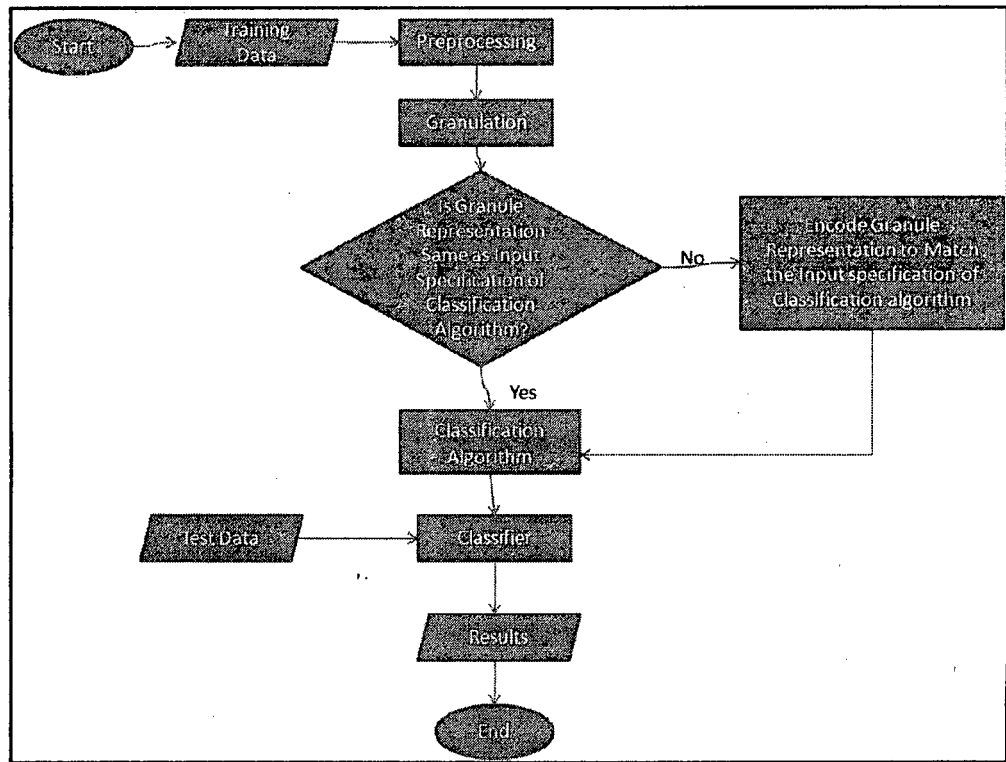


Figure4.1 Framework of Classification using Granulation

4.2. Granulation

The algorithm to construct the granulation tree is applied to the training data to obtain the granulated view of training data. Each leaf node of this granulation tree represents one granule. Statistical heuristic is used to find the significance of attributes used to construct the granulation tree. The number of attributes to decompose the dataset are limited in order to limit the size of granulation tree and hence the performance issues. The average entropy of the attributes is calculated and that attributes which has the entropy less than that of the average entropy are chosen for the decomposition. The subsets generated from this decomposition are further decomposed until all the attributes that satisfy the entropy constraints are used for decomposition. The collective traversal of these subsets with null as entropy value gives the granulation tree [19]. The detailed description of the algorithms is given in the Figure4.2.

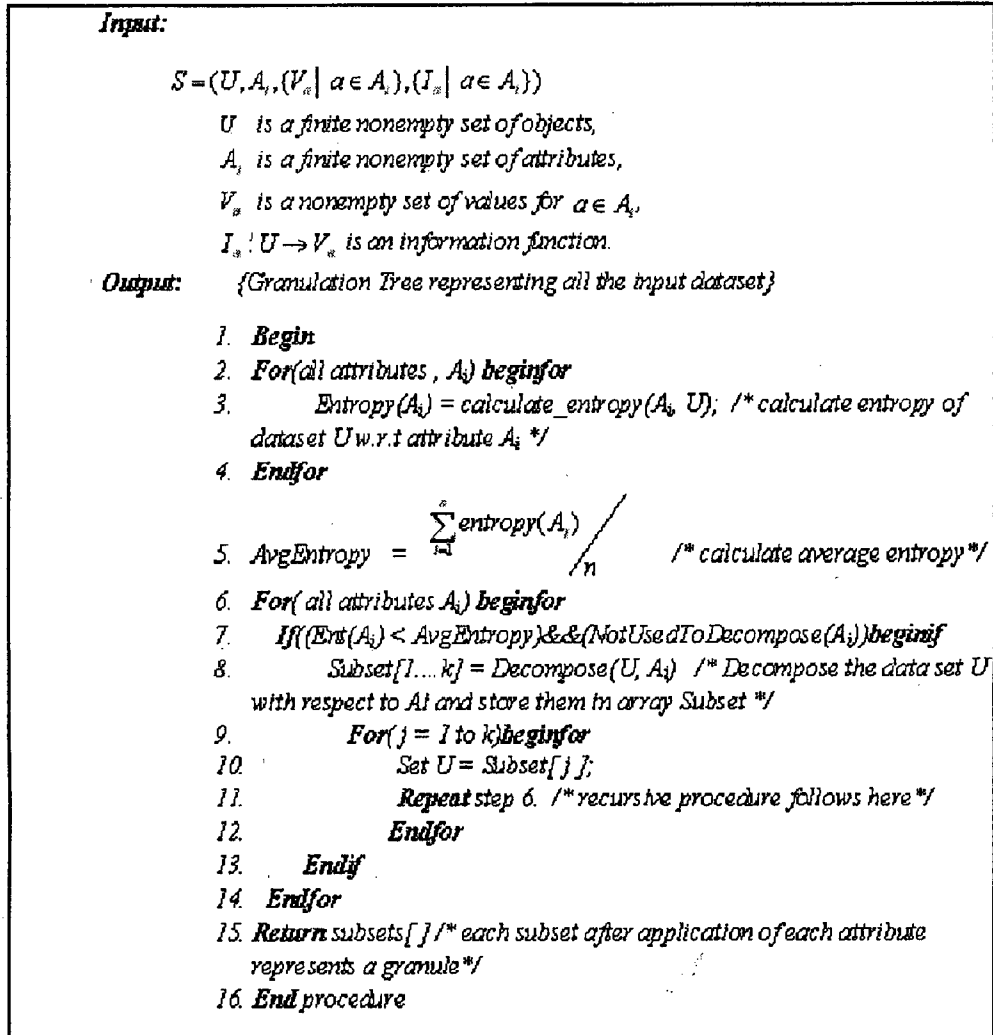


Figure4.2 Granulation

4.3. Granule Representation

In this dissertation the granules are represented by the random selection of data tuples from the leaf nodes of the granulation tree [19]. Number of tuples may be decided for a given data based on size of granule, quality of granule etc. The set of tuples representing all the granules is the representation of granulated view of the entire training dataset. Further encoding is beyond the scope of this dissertation. The algorithm for representation of granules is presented in Figure4.3.

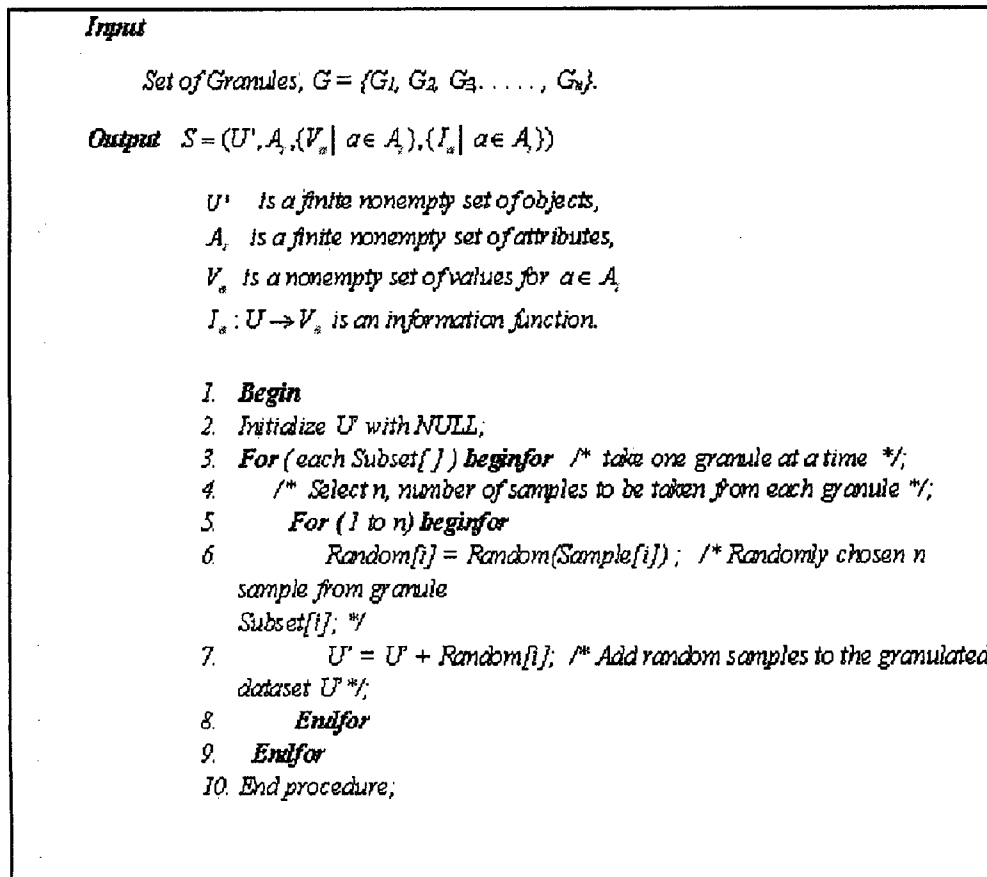


Fig4.3 Granule Representation

4.4. Implementation Details

All the algorithms explained above are implemented on JAVA programming platform. The data set is directly linked from the text file format. The training data is randomly sampled from granules and is used for the granulation purpose. Data is used for the testing purpose which is saved in separate file. The dataset saved for the granulation purpose is input to the granulation tree model. The granulated data is then represented by selecting the data tuples randomly from the leaf nodes of the granulation tree.

CHAPTER 5

EXPERIMENTS & RESULTS

An important goal of this dissertation is to study the performance of classification algorithms in granular computing environment. In order to achieve this, experiments are performed on a number of datasets by applying two classification algorithms namely ID3 and Naïve Bayesian classification algorithms as described in chapter4. Instead of directly applying data to train the classifier, the training data is granulated. Granules are formed using attribute oriented granulation model, Granulation tree [18]. In this chapter the classifiers are evaluated using various performance parameters. The granulated dataset is constructed using random sampling of tuples from granules. Thus the data reduction is achieved by the tuples that represent all the granules.

5.1 System Description

All the algorithms implemented so far have been coded in the JAVA programming platform. The system configuration is given as:

- **Hardware Used:** All the experiments are carried out on the computer with configuration as,
 - Main Memory:** 1GB (DDR).
 - CPU:** Intel Pentium 4(1.8 GHz with 1 MB L2 cache memory).
- **Software Used:**
 - OS:** windows xp (professional edition).
 - Software:** Weka software for Data Mining.
 - Platform:** Java 1.6(jdk-6u3-windows-I 586-p).

5.2 Dataset Description

Four datasets namely Buycomputer, Balancescale, Nursery and Poker are taken into consideration to accomplish the experiment. In order to observe the uniformity of the results all size of databases of variable size are considered. Buy computer is very small dataset which contains only 14 tuples. Balancescale and Nursery are medium sized datasets. Poker with 25010 tuples is a good sized dataset [23].

The experiments were considered mainly to observe the impact of Granular computing approach, therefore no preprocessing techniques, like discretization, filling null values, data normalization etc is taken care of. Therefore all the dataset used in the experiment are nominal dataset. Each dataset contains a class attribute which describes the associated class label on each tuple. The various dataset used are given in table 5.1.

Table 5.1: Data Description

Data Sets	No. of instan.	No. of Att.	Type of att.	No.of class values
<i>Buy comp</i>	14	5	<i>Nominal</i>	2
<i>Balance scale</i>	625	5	<i>Nominal</i>	3
<i>Nursery</i>	7937	9	<i>Nominal</i>	5
<i>Poker</i>	25010	11	<i>Nominal</i>	9

5.3 Evaluation Parameters

The performance of classifiers is evaluated using a number of parameters. However in this dissertation only accuracy and precision have been considered. One of the useful tools for analyzing how well the classifier can recognize tuples of different classes is confusion matrix. A confusion matrix for two classes shown as:

Table 5.2: Confusion Matrix

		Actual class	
		C1	C2
Predicted Class	C1	<i>True positive</i>	<i>False negative</i>
	C2	<i>False positive</i>	<i>True negative</i>

True positive is the number of tuples actually belonging to class C_1 and are predicted of class C_1 . True negative is the number of tuples actually belonging to class C_2 and are predicted as C_2 . Similarly false positive is the tuples of class C_1 and are predicted as of C_2 . False negative are the tuples of class C_2 which are predicted as C_1 .

Given m classed a confusion matrix is a table of size m by m . Each entry $CM_{i,j}$ in the first m rows and m columns indicates the number of tuples of class i that were labeled by the classifier as class j . For a classifier to have a good accuracy, most of the tuples would be represented also the diagonal of the confusion matrix that is from $CM_{1,1}$ to entry $CM_{m,m}$ with the rest of entries being close to zero.

Using the terms of confusion matrix, accuracy and precision can be defined as follows:

$$accuracy = \frac{t_pos + t_neg}{|D|}$$

And,

$$precision = \frac{t_pos}{(t_pos + f_pos)}$$

Here t_pos is total no. of positive tuples and t_neg is total no. of negative tuples.

The accuracy of a classifier on a given test data set is the percentage of test set tuples that are corrected classified by the classifier. This is also known as recognition rate of the classifier that is how well the classifier recognizes tuples of the various classes [9, 10].

5.4 Description of Experiment

In order to evaluate the impact of granular computing on classification, the classifiers are induced from each of four datasets without granulation and with granulation. The performance of classifiers without granulation is observed to be

compared with the classifiers induced after granulation. Data tuples for experiment is first sampled from granules. The classification algorithms are then applied on that granulated data. The data from the data granules are sampled under following schemes:

- One data element per granule: Under this scheme only one data instance is taken chosen from one granule. In this way the number of data in the granule data set is equal to the number of granules. Advantage of this sampling scheme is that when the original dataset is very large and the no. of attributes in the data set are less, large no. of granules are formed and hence it reduced the size of granulated data up to significant amount.
- Two data elements per granule: Under this scheme two data instance are chosen from one granule. This kind of scheme is used when the size of original dataset is moderated. In this way an appropriated number of granules are constructed.
- Three data elements per granule: Under this scheme three data instance are chosen from one granule. This kind of scheme is used when the size of original dataset is small. Choosing three data instances per granule make the granulated data sufficient to train a classifier.

5.4.1 Results of classification without granulation

The data is subject under two classification algorithms ID3 and Naïve Bayesian. Results of two classification algorithms are observed with 10 fold cross validation approach. Table5.4 shows the average accuracy results.

Table 5.3 Accuracy without granulation

Data	classifier	10 fold cross validation	
		ID3	Naïve Bayes'
	Buy comp	85.7	57.14
	Balance Scale	38.9	91.2
	Nursery	98.7	88.9
	Poker	39.4	49.06

The precision without granulation is given as in the table,

Table4.4 Precision without Granulation

Data set	classifier	10 fold cross validation	
		ID3	Naïve Bayes'
	Buy comp	.864	.642
	Balance Scale	.448	.926
	Nursery	.992	.913
	Poker	.463	.541

5.4.2 Granulation results

The input data is first granulated and then various classification algorithms are applied on this granulated data. The results of granulation are shown in this table as:

Table5.5 Granulation

Data Sets	No.of tuples	No. of Granules	Granule based Dataset (% of original Dataset)		
			1 Ob/Gr	2 Ob/Gr	3 Ob/Gr
<i>Buy comp</i>	14	5	5(35.5)	8(57.1)	10(71.4)
<i>Balance scale</i>	625	115	115(18.4)	217(34.7)	319(51.4)
<i>Nursery</i>	7937	3288	3288(41.4)	4805(60.5)	6226(78.4)
<i>Poker</i>	2501	9246	9246(36.9)	13755(54.9)	18427(73.6)

5.4.3 Results of classification using granulation

The results of granulation have been recorded on four classifiers, ID3, Naïve Bayesian classifiers. First the granulated data is used to train the classifier and then the test data is applied on that classified to measure the accuracy. Table5.6 shows the results.

Table 5.6 Accuracy with granulation

Data ↓ sets (% reduced)		Classifier →		Training		Test	
		ID3	NaïveBayes'	ID3	NaïveBayes'		
Buy comp	1 Ob(35.7)	20	20	66.6	66.6		
	2 Ob(57.1)	39	75	25	75		
	3 Ob(71.4)	75	100	75	75		
Balance Scale	1 Ob(18.4)	48.6	76.5	67.9	88.77		
	2 Ob(34.7)	71.4	84.3	55.6	83.9		
	3 Ob(51.4)	84.01	88	59.65	84.2		
Nursery	1 Ob(41.4)	96.4	92.9	95.1	88.6		
	2 Ob(60.5)	98.4	92.7	94.1	89.1		
	3 Ob(78.4)	98.8	93.0	94.0	89.2		
Poker	1 Ob(36.9)	36.2	46.7	53.8	49.67		
	2 Ob(54.9)	69.2	48.19	57.56	49.3		
	3 Ob(73.6)	78.8	49.1	82.77	51.04		

Values in table5.6 for without granulation the average accuracy has been considered while for the experiments after granulation the best classification accuracy has been considered for the comparison of results.

Table 5.7 Precision with granulation

Data ↓ Classifier → sets (to % reduced)		Training		Test	
		ID3	NaïveBayes'	ID3	NaïveBayes'
Buy comp	1 Ob(35.7)	.11	.11	.643	.672
	2 Ob(57.1)	.436	.861	.33	.784
	3 Ob(71.4)	1	1	1	.873
Balance Scale	1 Ob(18.4)	.712	.737	.779	.842
	2 Ob(34.7)	.793	.768	.732	.773
	3 Ob(51.4)	.889	.881	.732	.738
Nursery	1 Ob(41.4)	.987	.936	.966	.892
	2 Ob(60.5)	.992	.932	.957	.891
	3 Ob(78.4)	.995	.935	.958	.895
Poker	1 Ob(36.9)	.528	.415	.673	.445
	2 Ob(54.9)	.824	.432	.686	.442
	3 Ob(73.6)	.894	.492	.657	.439

5.4.4 Comparison

The results of accuracy without granulation and with granulation are given here in Table 5.8. The accuracy results with granulation are determined by taking the average of accuracies with one, two and three data tuples per granulation.

Table 5.8 Accuracy Comparison

Data	Classification Algorithm	Accuracy (Without Granulation)	Accuracy (With Granulation)	Best Results (Type of Data)
<i>Buy</i>	<i>DT</i>	85.7	55.5	75(71.4%)
	<i>Bayesian</i>	57.14	72.2	75(71.4%)
<i>Balance scale</i>	<i>DT</i>	38.9	61	67.9(18.4%)
	<i>Bayesian</i>	91.2	85.4	88.7(18.4%)
<i>Nursery</i>	<i>DT</i>	98.7	94.2	95.1(41.4%)
	<i>Bayesian</i>	88.9	88.7	89.2(78.4%)
<i>poker</i>	<i>DT</i>	39.4	64.7	82.7(73.6%)
	<i>Bayesian</i>	49.6	50.01	51.4(73.6%)

In Buy computer dataset, the Bayesian classifier gives better accuracy with granulation and compared to ID3. For intermediate size datasets, like balance scale, DT prove itself a good classifier but and the accuracy of Bayesian is reduced insignificantly but the size of training dataset is reduced significantly. In good sized dataset Nursery, there is insignificant reduction in accuracy but the size of training dataset is reduced significantly. Poker is a large dataset with large number of attribute values. Hence the classifiers could not perform well but still a great reduction in dataset.

The result of comparison of precision is given in table5.8. The precision results are analogous to that of the accuracy results. The precision with granulation on Buy computer dataset is better than that of without granulation. But it is difficult to reach to any conclusion since buy computer is a small dataset. On intermediate dataset balance scale, ID3 performs better than that of Bayesian classifiers. Both ID3 and Bayesian

classifiers performs well on Nursery dataset. The precision performance on poker dataset is not so good for both ID3 and Bayesian classifiers. The cause behind this appears the large no. of attribute value and class values.

Table5.9 Precision Comparison

Data	Classification Algorithm	Precision(Without Granulation)	Precision(With Granulation)	Best Precision (Type of Data)
<i>Buy</i>	<i>ID3</i>	.864	.657	1(71.4%)
	<i>Bayesian</i>	.642	.773	.873(71.4%)
<i>Balance scale</i>	<i>ID3</i>	.448	.748	.779(18.4%)
	<i>Bayesian</i>	.926	.784	.842(18.4%)
<i>Nursery</i>	<i>ID3</i>	.992	.960	.967(41.4%)
	<i>Bayesian</i>	.913	.893	.895(78.4%)
<i>poker</i>	<i>ID3</i>	.463	.672	.686(54.9%)
	<i>Bayesian</i>	.541	.442	.445(36.9%)

5.4.5 Analysis

The classification using granulation is analyzed on the account of two classification algorithms ID3 and Naïve Bayesian. The results show that the classification using granulation performs better than without granulation except few results. The significance of granulation process is that the training dataset is reduced significantly.

CHAPTER 6

CONCLUSION AND FUTURE WORK

So far in this dissertation we have studied how the granular computing can be applied in the classification problems. Classification and the issues regarding classification are discussed in detail. Various type of classification algorithms are explained with appropriate detail in chapter2. Granular computing and the components of granular computing are explained in chapter3. Granules and the relationship between granules is explained with proper hierarchy. Various models and frameworks of granulation are explained with an appropriate detail in this dissertation. The proposed framework of classification using granulation is explained in chapter4. The basic idea of granulation is incorporated from granulation tree model of granulation.

The various experiments on the datasets are performed. The accuracy and the precision results by the classification algorithms ID3 and Bayesian classification algorithms are observed. A discussion on these results is given in chapter5 of the dissertation. The attribute selection measure to construction of granules shows its magic. It has been observed that the data in the granules are uniformly divided.

Another advantage of this model is observed so far is that the model works very effectively even when the dataset is very large. Hence the scalability issue of various data mining algorithms can be taken care by using the granular computing approach.

FUTURE WORK

The data set so far chosen are having only the nominal attributes. The proposed models don't work on the continuous valued attributes. There was no missing value in the database. But this is not the case in real time databases. The attributes in the databases generally have mixed attribute with both nominal and continuous values. Missing values are also present in the real time data base. Keeping these problems in mind, my future goal will be to design a model which can cope up with these problems of the ordinary databases.

In this model of classification using granulation, the data is selected randomly from the granules. It could have been better if some heuristic approach for the selections of data from granule be applied. My future work will be concentrated on this issue.

References

1. Y. Y. Yao, **Granular Computing**, Computer Science, Vol.31,1-5, Proc. 4th Chinese National Conf. on rough sets and soft computing, 2004.
2. Y. Y. Yao, **A partition model of Granular Computing**, LNCS Transactions on Rough Sets, Vol.1, Page(s):232-253, 2004.
3. T. Y. Lin, **Data Mining and Machine Oriented Modeling: A Granular Computing Approach**, Journal of Applied Intelligence, Kluwer, Vol. 13, No 2, September/October, 20, pp. 113-124.
4. Jing Tao Yao, Senior Member, **IEEE Information Granulation and Granular Relationships**.
5. Pawlak, Z. **Rough sets**, International Journal of Computer and Information Sciences, 11, 341-356, 1982.
6. J.T. Yao and Y.Y. Yao, **Induction of classification rules by granular computing**, Proceedings of the Third International Conference on Rough Sets and Current Trends in Computing, Lecture Notes in Artificial Intelligence, pp33 1-338, 2002.
7. Pedrycz, W., Smith, M. H. (1999) **Granular correlation analysis in data mining**, Conf of the North American Fuzzy Information Processing Society (NAFIPS), New York
8. Zadeh, L. A. (1997) **Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic**, Fuzzy Sets and Systems
9. J. Han & Kamber, M, **Data Mining: Concepts and Techniques** San Francisco: Morgan Kaufmann, 2001.
10. W. Pedrycz, **Granular Computing: An introduction**, IEEE Transactions, Page(s): 1349-1354, 2001.
11. Y.J. Tao, **Information Granulation and Granular Relationships**, Proc. Granular Computing, 2005 IEEE International Conference July 2005 Page(s): 326-329, Vol. 1.

12. Y. Y. Yao, **Perspectives on Granular Computing**, Proc. IEEE Conf. on Granular Computing, Vol.1, Page(s): 85-90, 2005.
13. Y. Y. Yao, **Granular Computing Basic Issues and Possible Solutions**, Proceedings of the 5th Joint Conference on Information Sciences, Page(s):1186-189, 2000.
14. Y. Y. Yao, **Granular Computing for Data Mining**, Proceedings of SPIE Conference on Data Mining, Intrusion Detection, Information Assurance, and Data Networks, 2004.
15. Pedrycz, W. Smith, M.H. (1999) **Granular correlation analysis in data mining**, proc, 18th Int Conf of the North American Fuzzy Information Processing society(NAFIPS), New York.
16. Zadeh, L.A. **Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic**, Fuzzy sets and System.
17. Yao, Y.Y. and Zhong, N. **Granular Computing using Information Tables**, in: **DatMining, Rough Sets and Granular Computing**, Lin, T.Y., Yao, Y.Y. and Zadeh, L.A. (Eds.), Physica-Verlag, Heidelberg.
18. T. Y. Lin, **Neighborhood Systems and Approximation in Database and Knowledge Base Systems**, Proceedings of the Fourth International Symposium on Methodologies of Intelligent Systems, Poster Session, Charlotte, North Carolina, October 12–15, pp. 75–86, 1989.
19. Girish Kumar Singh, and Sonajharia Minz **Granulation using Clustering and Rough Set Theory & its Tree Representation**; World Academy of Science, Engineering and Technology 25 2007
20. Y. Y. Yao, N. Zhong, **Potential Applications of Granular Computing in Knowledge Discovery and Data Mining**, in: Proceedings of World Multi conference on Systemic, Cybernetics and Informatics, Computer Science and Engineering, Orlando, 1999: 573-580.
21. Z. Pawlak, **Rough Sets-Theoretical Aspects of Reasoning about Data**, Kluwer Academic Publishers, Boston, London, Dordrecht, 1991.

22. Y.Y. Yao, J. T. YAO, **Granular computing as a Basis for Consistent Classification Problems**, in: Proceedings of PAKDD2002 Workshop entitled Towards Foundation of Data Mining, Communications of Institute of Information and Computing Machinery, 2002, 5(2): 101-106.
23. <http://archive.ics.uci.edu/ml/machine-learning-databases/>

