CHAPTER 3

SCOPE OF RESEARCH WORK

3.1 OBJECTIVES OF THE STUDY
Purpose of this work was to diagnose the state of efficiency in itself and to trace out the factors responsible for lower or higher efficiency in discharging various operations and activities of analysis such as Knowledge Management and data mining quality and statement[3]. Bearing in mind that confusion surrounds the criteria by which the performance of algorithms, it is guided by information requirements and suitable database(s) to achieve improved KDD (Knowledge Discovery in Databases) processing.
To justify the objective of the work, different processes for supervised learning were analyzed with experimental results. A few of the accepted and practical indices permitting to quantitative and qualitative dimensions were applied to appraise the performance of the corporation[6]. In brief, the objective of the study was:

1. To just expose data processing operations with its objectives to detect and examine reasons thereof.
2. To assess their quality performance through structural analysis and empirical measurement.
3. To study and analyze the quality of data processing based on quality, services and business strengths.

3.2 THE PROBLEM
In Knowledge Discovery, Data Mining is relative new concept. Now the whole world is globalize, liberalization polices are adopted by government, so in the changing scenario, the Apex body of industrial, organization, Business organization, Education organization, Engineering Institutions, Web designers
and managers, problems related to Supervised Learning, Classification, Regression and their Fusion Supervised learning process is applied with certain modifications for different problems and their environments[9]. This type of customization demands big deal of domain knowledge and experience in implementing suitable process for problem at hand. Different problems related to supervised learning, classification, regression and their fusion are discussed as follows.

1. Data mining researches has explored sufficiently large number of algorithms for supervised learning. Different algorithms classify data according to their own criteria and usually differ in their results. Different algorithms accompany many parameters to be adjusted for the improvement of their performance. Supervised processing involves many processes as well, however finding a suitable process is a difficult task. These processes target to optimize size of the model with respect to learning. All these factors create difficulty in applying a suitable process for problem at hand.

2. The need for classification task appears in a variety of applications, from industry to research and from common life to professional life; where there is need to select one among many available options. Training of computers for such decision making tasks has gained huge attention. Different applications use data mining solutions for their classification problems. Varying applications demand varying modification of supervised learning process. These applications also demand variety of metrics for their analysis of data. Varying applications, different mining processes and need for different metrics makes classification process complex.

3. Regression based algorithms can be used to classify quantitative data. Qualitative data need to be coded into quantitative data. Experience with regression based algorithms for classification is not so much
impressive as compared to the techniques like decision trees and neural networks etc.

3.3 SCOPE OF THE RESEARCH WORK
There are many data mining technique Method but we need to select best data mining technique for data processing. This work primarily includes investigations of supervise learning techniques and algorithms [5]. Classifications tasks are included in the majority of applications of supervise learning. This thesis concentrates upon classification tasks through supervised learning. We have included applications from various domains for investigations. The study covers operation efficiency of knowledge discovery in data processing quality in data mining technique. In particular the empirical evaluations of following algorithm were performed.

1. Classification algorithms
   a. K-nearest neighborhood
   b. Naive Bayes
   c. Decision Tree
   d. Decision Stump
   e. Rule Induction

2. Decision tree algorithms
   a. BF Tree
   b. FT Tree
   c. J48 Tree
   d. LAD Tree

3. Neural network algorithm
   a. Multilayer Perceptron
   b. Radial Basis Function

4. Association Rule Mining Algorithms
   a. Apriori
   b. FP-Growth
3.4 KNN CLASSIFICATION
A k-nearest neighbor classifier is a simple data classifier, where the category (class label) of a data item is determined by using a majority vote of its neighbors, assigning the data item to the category most common among its k nearest neighbors. The input to the algorithm is a set of tuples, where one of the attributes is a class label and the other attributes are the features of the given data. While some of the tuples have known class labels, others have unknown class labels, and the task is to label those tuples with unknown class labels. The class label for any tuple \( t \) is determined with the following procedure:

1. Measure the distance of the tuple \( t \) to each of the labelled tuples in the dataset.
2. Find the set \( S \) of the k-nearest neighbors of \( t \) (i.e. \( k \) tuples with the smallest distance to \( t \)).
3. Find the majority class label in the set \( S \). If two or more classes have the majority (i.e. the same number of occurrences in the set), decrease \( k \) by 1 until we find the majority class label.

Although various distance metrics could be used to measure the distance between two data items in KNN, one of the most important metric is the Euclidean distance. Given two tuples \( T1 \) and \( T2 \) with \( n \) features (where \( T1_x \) represents the \( x^{\text{th}} \) feature value of \( T1 \)), the Euclidean distance between \( T1 \) and \( T2 \) is calculated as follows:

\[
\sqrt{(T1_1 - T2_1)^2 + (T1_2 - T2_2)^2 + (T1_3 - T2_3)^2 + \cdots + (T1_n - T2_n)^2}
\]

3.5 NAÏVE BAYES
In simple terms, a naive bayes classifier assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature. For example, a fruit may be considered to be an apple if it is red, round, and about 4” in diameter. Even if these features
depend on each other or upon the existence of the other features, a naive bayes classifier considers all of these properties to independently contribute to the probability that this fruit is an apple.

Depending on the precise nature of the probability model, naive bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for naive bayes models uses the method of maximum likelihood; in other words, one can work with the naive bayes model without believing in Bayesian probability or using any Bayesian methods.

3.6 DECISION TREE

Decision trees are a way of representing a series of rules that lead to a class or value. The first component is the top decision node, or root node, which specifies a test to be carried out. The results of this test cause the tree to split into branches, each representing one of the possible answers. Depending on the algorithm, each node may have two or more branches. For example, CART generates trees with only two branches at each node. Such a tree is called a binary tree. When more than two branches are allowed it is called a multi way tree. A decision tree partitions the input space of a data set into mutually exclusive regions, each of which is assigned a label, a value or an action to characterize its data points. The decision tree mechanism is transparent and we can follow a tree structure easily to see how the decision is made. A decision tree is a tree structure consisting of internal and external nodes connected by branches. An internal node is a decision making unit that evaluates a decision function to determine which child node to visit next. The external node, on the other hand, has no child nodes and is associated with a label or value that characterizes the given data that leads to its being visited. However, many decision tree construction algorithms involve a two–step process. First, a very large decision tree is grown. Then, to reduce large size and overfitting the data, in the second step, the given tree is pruned. The
pruned decision tree that is used for classification purposes is called the classification tree.

### 3.7 DECISION STUMP

A decision stump is a machine learning model consisting of a one-level decision tree. That is, it is a decision tree with one internal node (the root) which is immediately connected to the terminal nodes (its leaves). A decision stump makes a prediction based on the value of just a single input feature. Sometimes they are also called 1-rules. Depending on the type of the input feature, several variations are possible. For nominal features, one may build a stump which contains a leaf for each possible feature value or a stump with the two leaves, one of which corresponds to some chosen category, and the other leaf to all the other categories. For binary features these two schemes are identical. A missing value may be treated as a yet another category.

### 3.8 RULE INDUCTION

Rule induction is one of the most important techniques of machine learning. Since regularities hidden in data are frequently expressed in terms of rules, rule induction is one of the fundamental tools of data mining at the same time. Usually rules are expressions of the form:

\[
if \ (attribute - 1, value - 1) \ and \ (attribute - 2, value - 2) \ and \ \ldots \ \\
and \ (attribute - n, value - n) \ then \ (decision, value)
\]

Some rule induction systems induce more complex rules, in which values of attributes may be expressed by negation of some values or by a value subset of the attribute domain. Data from which rules are induced are usually presented in a form similar to a table in which cases (or examples) are labels (or names) for rows and variables are labelled as attributes and a decision. In different words, the decision value is assigned by an expert to each case. Attributes are independent variables and the decision is a dependent variable.
3.9 BF TREE

Best First Tree Standard algorithms for the top-down induction of decision trees expand nodes in depth-first order in each step using the divide-and-conquer strategy. A fixed order is used to expand nodes (normally, left to right) the decision trees. In Best-first decision trees the selection of best split is based on boosting algorithms, which is used to expand nodes in best-first order instead of a fixed order. This algorithm uses the both the gain and index in calculating the best node in tree grown phase of the decision tree. This method adds the “best” split node to the tree in each step. The best node is the node that maximally reduces impurity among all nodes available for splitting (i.e. not labelled as terminal nodes). Although this results in the same fully grown tree as standard depth-first expansion, it enables us to investigate new tree pruning methods that use cross-validation to select the number of expansions. Both pre-pruning and post-pruning can be performed in this way.

3.10 FT TREE

FT combines a standard univariate DT, such as C4.5, with linear functions of the attributes by means of linear regressions. While a univariate DT uses simple value tests on single attributes in a node, FT can use linear combinations of different attributes in a node or in a leaf. In the constructive phase a function is built and mapped to new attributes. A model is built using the constructor function. This is done using only the examples that fall at this node. Later, the model is mapped to new attributes. The constructor function should be a classifier or a repressor depending on the type of the problem. In the former the number of new attributes is equal to the number of classes, in the later the constructor function is mapped to one new attribute. Each new attribute is computed as the value predicted by the constructed function for each example. In the classification setting, each new attribute value is the probability that the example belongs to one class given by the constructed
model. The merit of each new attribute is evaluated using the merit-function of the univariate tree, and in competition with the original attributes.

3.11 J48 TREE

It builds the decision tree from labelled training data set using information gain and it examines the same that results from choosing an attribute for splitting the data. To make the decision the attribute with highest normalized information gain is used. Then the algorithm recurs on smaller subsets. The splitting procedure stops if all instances in a subset belong to the same class. Then the leaf node is created in a decision tree telling to choose that class.

3.12 LAD TREE

Logical Analysis of Data is the method for classification proposed in optimization literature. It builds a classifier for binary target variable based on learning a logical expression that can distinguish between positive and negative samples in a data set. The basic assumption of LAD model is that a binary point covered by some positive patterns, but not covered by any negative pattern is positive, and similarly, a binary point covered by some negative patterns, but not covered by positive pattern is negative. The construction of Lad model for a given data set typically involves the generation of large set patterns and the selection of a subset of them that satisfies the above assumption such that each pattern in the model satisfies certain requirements in terms of prevalence and homogeneity.

3.13 MULTILAYER PERCEPTRON

Multilayer Perceptron classifier is based upon backpropagation algorithm to classify instances. The network is created by an MLP algorithm. The network can also be monitored and modified during training time. The nodes in this network are all sigmoid except for when the class is numeric in which case the output nodes become unthresholded linear units. The backpropagation neural network is essentially a network of simple processing elements working
together to produce a complex output. The backpropagation 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. An example of a multilayer feed-forward network is shown in Figure 3.1.

![Diagram of a multilayer feed-forward network](image)

Figure 3.1: Multilayer feed-forward network

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 inputs pass through the input layer and are then weighted and fed simultaneously to a second layer of “neuronlike” units, known as a hidden layer. The outputs 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. At the core,
backpropagation is simply an efficient and exact method for calculating all the 
derivatives of a single target quantity such as pattern classification error with 
respect to a large set of input quantities such as the parameters or weights in a 
classification rule. To improve the classification accuracy we should reduce 
the training time of neural network and reduce the number of input units of the 

network.

3.14 RADIAL BASIS FUNCTION
The Radial Basis Function (RBF) is another popular architecture used in 
ANN. The RBF, which is multilayer and feed-forward, is often used for strict 
interpolation in multi-dimensional space. The term “feed-forward” means that 
the neurons are organized as layers in a layered neural network.
The RBF network comprises three layers, i.e. input, hidden and output. The 
input layer is composed of input data. The hidden layer transforms the data 
from the input space to the hidden space using a non-linear function. The 
output layer, which is linear, yields the response of the network. The argument 
of the activation function of each hidden unit in an RBF network computes the 
Euclidean distance between the input vector and the center of that unit. In the 
structure of RBF network, the input data, $x$, is a p-dimensional vector, which is 
transmitted to each hidden unit. The activation function of hidden units is 
symmetric in the input space, and the output of each hidden unit depends only 
on the radial distance between the input vector, $x$, and the center for the hidden 
unit. Each node in the hidden layer is a p-multivariate Gaussian function, 
given as follows:

$$G(x', x_i) = \exp \left[ -\frac{1}{2\sigma_i^2} \sum_{k}^p (x_k - x_k')^2 \right]$$

where: $x_i$ is the mean (center) and $i$ is the variance (width). These functions are 
referred to as radial basis functions. Finally, a linear weight is applied to the 
output of the hidden nodes to obtain:
The problem with this solution is that it may lead to a very large hidden layer. Thus, the solution should be approximated to reduce the number of PEs in the hidden layer and cleverly position them over the input space regions. This entails the need to estimate the position of each radial basis function and its variance, as well as to compute the linear weights, $w_i$. An unsupervised technique, known as the $k$-nearest neighbor rule, is used to estimate the mean and the variance. The input space is first discretized into $k$ clusters and the size of each cluster is obtained from the structure of the input data. The centers of the clusters give the centers of the RBFs, while the distance between the clusters provides the width of the Gaussians. The output weights are obtained through supervised learning. Therefore, the error correction learning described earlier in the MLP is employed.

3.15 APRIORI ALGORITHM

Apriori is the best-known algorithm to mine association rules. It uses a breadth-first search strategy to counting the support of itemsets and uses a candidate generation function which exploits the downward closure property of support. Apriori uses a "bottom up" approach, where frequent subsets are extended one item at a time (a step known as candidate generation), and groups of candidates are tested against the data. The algorithm terminates when no further successful extensions are found.

3.16 FP-GROWTH

FP-growth (frequent pattern growth) uses an extended prefix-tree (FP-tree) structure to store the database in a compressed form. FP-growth采用 a divide-and-conquer approach to decompose both the mining tasks and the databases. It uses a pattern fragment growth method to avoid the costly process of candidate generation and testing used by Apriori. FP-Growth
adopts a divide-and-conquer strategy as follows. First, it compresses the database representing frequent items into a frequent-pattern tree or FP-tree, which retains the itemsets association information. It then divides the compressed database into a set of conditional databases. Each associated with one frequent item and mines each such database separately.