CHAPTER 1

INTRODUCTION

1.1. Background of the Study

With the rapid increase in the number of computers connected to the Internet and the emergence of a range of mobile computational devices which might soon be equipped with Mobile Internet Protocol (IP) technology, the Internet is converging to a more dynamic, huge, fully distributed Peer-to-Peer (P2P) overlay networks containing millions of nodes typically for the purpose of information distribution and file sharing [1, 2, 3]. Such fully distributed systems generate large amount of data. Analyzing this data can be interesting from both scientific and business purpose. Among other applications, this environment is a natural target for distributed data mining [4]. The motivation behind P2P data mining includes the optimal usage of available computational resources, privacy and dependability by eliminating critical points of service [2].

Two constraints were adopted the first was all peers were allowed to hold as few data called "Expertise" and the second didn't have practically a limited number of nodes [1, 2]. Basically, the only requirement was that each peer could communicate directly with their Super-Peer which was the parent of the domain. Furthermore, important aspects, the first was data privacy; the second was the dynamic nature of the underlying network i.e. peers can leave the overlay network and new peers can join it.

The communication architecture are classified in two types, that is, Client-Server architecture and P2P architecture. The former one holds the name after the way it allows to share the information, that is, the communication nodes are connected with each other through a central powerful node, called the server. All the sources of information are contained in central server and other nodes are called client, they are interested to access information, communicate with the central server. This makes the flow of information unidirectional. ARPANET [5] was the dominant client-server technology during 80’s.

Next category is the P2P which is completely different from above concept. Here, no any single communication nodes are assigned as the central server. The sources of information are distributed among all the participant nodes in the network. Each peers are both either client or server, depending on the role whether they are searching or serving (consuming or supplying) the information within the network. The introduction of Napster [6] in 90's had followed the concept of P2P networking. The later developments include SIP [6], Gnutella [7] and so on. However, the early versions were somehow mixture of P2P and the client-server model because some of the functions like peer and resource discovery were made with the help of central server or a number of server pools. Though, these pioneer approaches, also termed as first generation P2P systems, have evolved to address the recent requirements.

To achieve goal this research suggest a system that used decision trees to extract Super-Peer that contains peers that are relevant with respect to a given query. This system is an unstructured P2P system based on an organization of peers around Super-Peers that is connected to Super-Super-Peer according to their semantic domains and also uses NBTree: The hybrid algorithm to extract Super-Peer that contains peers with relevant data that respect a given query. The advantages of this model are the robustness in Queries Answering time, Minimum Query Processing and Scalability issues.

1.2.Statement of Problem

With the advancements in the available technology the number of computers connected to the internet is increased along with which the scope of communication is grown up to the complexity with ease of end users and the distribution of information for communication has been increased tremendously as well. Because of which a challenging problem in unstructured P2P system is how to locate peers that are relevant with respect to a given query with

- Minimum query processing
- Minimum answering time

1.3.Objective of the Study

The objective of this research is

- To use NBTree: The Hybrid Algorithm for the optimization of global index in proposed P2P system architecture.

1.4.Limitations of the Study

Limitations of this research are:

- This had done in context of unstructured P2P system architecture.
- This had done by the implementation of NBTreee: the hybrid algorithm in Weka version 3.7.10.
- This research compared the accuracy, precision, recall, f-measure of classification, tree size for optimization of query processing and time taken to build model.
- Data set was comprised of seven attributes
    - Super Peer
    - Query
    - ComponentW1
    - ComponentW2
    - ComponentW3
    - ComponentW4
    - Peer

1.5.Structure of the Report

This report is organized in five chapters including the following chapters.

- Chapter 1 ***"Introduction"*** explains the Background of the research, Statement of problems, Objectives of the study and Scope and Limitations of the study.
- Chapter 2 ***"Literature Review"*** describes the various concepts P2P networks, P2P architectures, machine learning and related works in our domain.
- Chapter 3 ***"Research Methodology"*** explains the framework and algorithm of the proposed P2P system architecture model.
- Chapter 4 ***"Experiment and Result Analysis"*** explains about experiments, results and context analysis.
- Chapter 5 ***"Conclusion and Future Work"*** explains the conclusion and future direction of the research.
- ***References***

CHAPTER 2

LITERATURE REVIEW

2.1. Peer-to-Peer (P2P) Networks

Peer-to-Peer (P2P) [8] is a distributed computing technology. This technology is used in direct interaction among the peers in order to exchange information without involvement of any mediator in between. A peer-to-peer network is a logical overlay network on top of a physical network [9]. Based on the functions that P2P does, we can define P2P broadly as self-organizing, decentralized distributed systems that consist of potentially non-trusted, unreliable nodes with symmetric roles.

Dana Moore and John Hebeler [9] define P2P as the action of mutually exchanging information and services directly between the producer and the consumer to achieve purposeful results. P2P exchanges messages and real-time information dynamically. The information can be shared between various parties regardless of the anonymous to highly trusted sources. The exchange of message is platform independent [9].

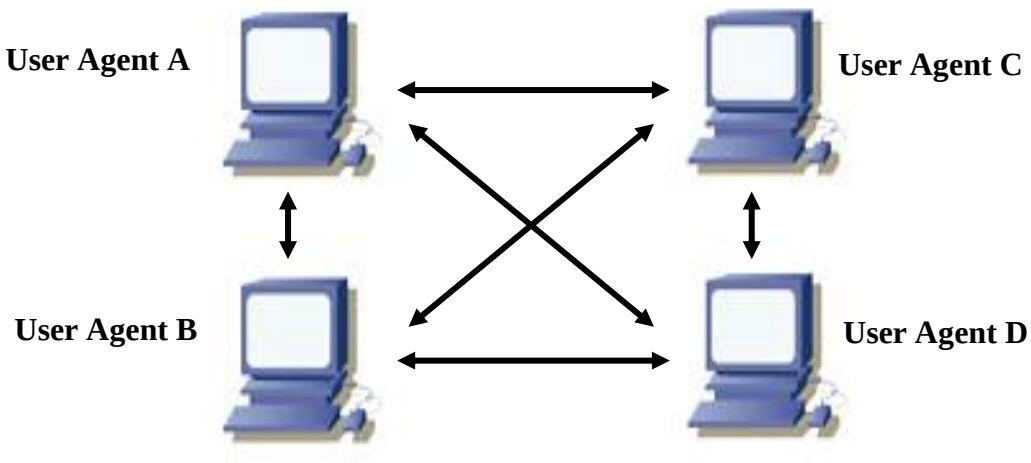

Figure 2.1: P2P Network

2.1.1. Importance of P2P Network

- P2P offers the information and services that may be important for the users.
- P2P offers less cost and saves time with respect to other network architecture in the sense that the requirement to operate P2P is possible in personal computers which has enough processing power, memory and disk storage [9].

- P2P provides publisher the total control over particular user to be allowed in accessing the resources and services.
- P2P offers a fully distributed symmetric architecture providing system reliability as well as integrity.

2.1.2. Dark Sides of P2P Network

- It is very difficult to predict anything about the P2P application since most of them lack any centralization.
- The integrity about particular resource is not possible at all the times.
- The resources are available to users as long as they are connected in the network.
- Since many P2P applications offer direct access to the user's information, the security becomes a crucial part to be discussed.

2.2. P2P Architecture

P2P architecture is a way to structure a distributed application such that many identical software modules can run on more than one machine [8]. These software modules then communicate with each other to perform the distributed action. These software modules can be accessed from another computer as well as allowing others to access its module at the same time. The software modules can be studied under following three layered structures as:

- Base overlay Layer
- Middleware Functions
- Application Layer

Base overlay layer functions to find out the other peers in the network called resource discovery [8]. This is performed either by discovering all the nodes in a network or a particular set of peers in network to perform the task. The layer also provides a mechanism to connect all the P2P nodes into a common network called the overlay network formation [3]. This layer also adds the functionality to multicast the message to all other peers in the network.

Middleware functions relate with the security issues for managing the secure communication among the peers. Because of the distributed nature of P2P, the security features like encryption, integrity, privacy, access control and authentication becomes a tough job. This is also responsible for distributed indexing of the resources in the network. This index is then used to locate the information in an efficient way. This layer also provides the controlling mechanism for delivering the messages effectively.

Application layer is the software component that allows user to gain various application related issues. The applications may be file sharing, queries routing, self-managing web sites, provide highly scalable instant messaging services and so on.

2.3. P2P Overlay Networks

The P2P overlay networks are broadly grouped into following classes as in figure 2.2.

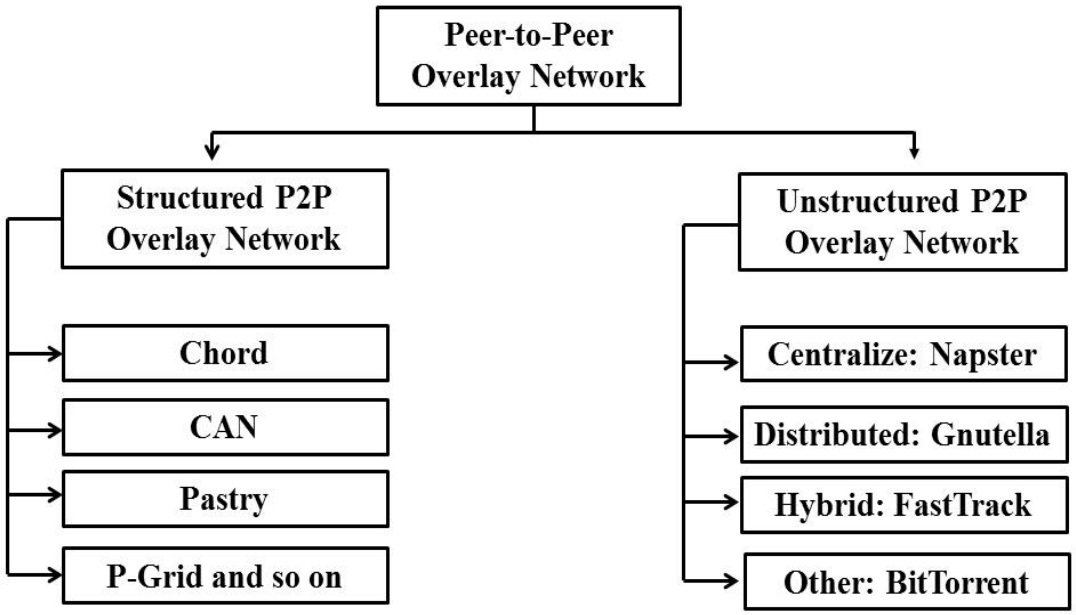

Figure 2.2: P2P overlay Networks distribution [1]

2.3.1. Unstructured P2P Overlay Network

The basic nature of unstructured P2P network is its unpredictability of the network topology. Any peers can join the network following some specific rules to be a part of the family in network. Connection between the peers is created randomly based on

availability. The peer that has higher availability sometimes may have multiple connections than those having less availability [8]. Apparently, this kind of network is suitable for the peers that join and leave the network frequently. The major drawback on unstructured network is to locate the resource efficiently. One of the most precise method of source identification in unstructured overlay is flooding method, in which query is sent to all the neighbouring nodes within a controlled radius. However, this topology is not an easy task to select since it uses a large part of bandwidth to flood the query. Search process in unstructured P2P networks are shown in *Figure 2.3 [10]*.

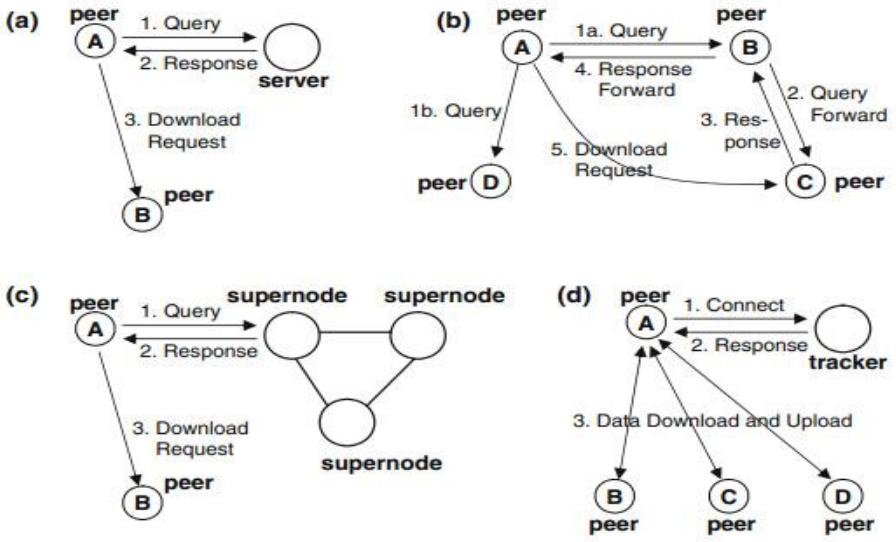

Figure 2.3: Search process in structured process in unstructured P2P networks

a) Napster b) Gnutella c) Kazaa and d) BitTorent

A. A Centralized Approach: Napster

It is based on the central database that maintains a central index and is accessible to all the nodes connected to this allowing sharing the information. The peers in the network use the central index so as to get response for the query they made during source discovery. The database then replies back the IP address of peer containing the information. Once the IP address is found, peers can exchange the message directly without involvement of central server.

The founder of Napster [6], Shawn Fanning, might not believe on success of his findings in short time. Since Napster [6] is basically targeted to share music in the form of MP3 files, it became a killer application and it got some legal issues on violation of copyright on music owners. As a result, it has to shut down the network in 2001. It is now in operation however with some new proprietary features of the source owner included. A general architecture showing the operation of Napster is shown in *figure 2.4 [6]*.

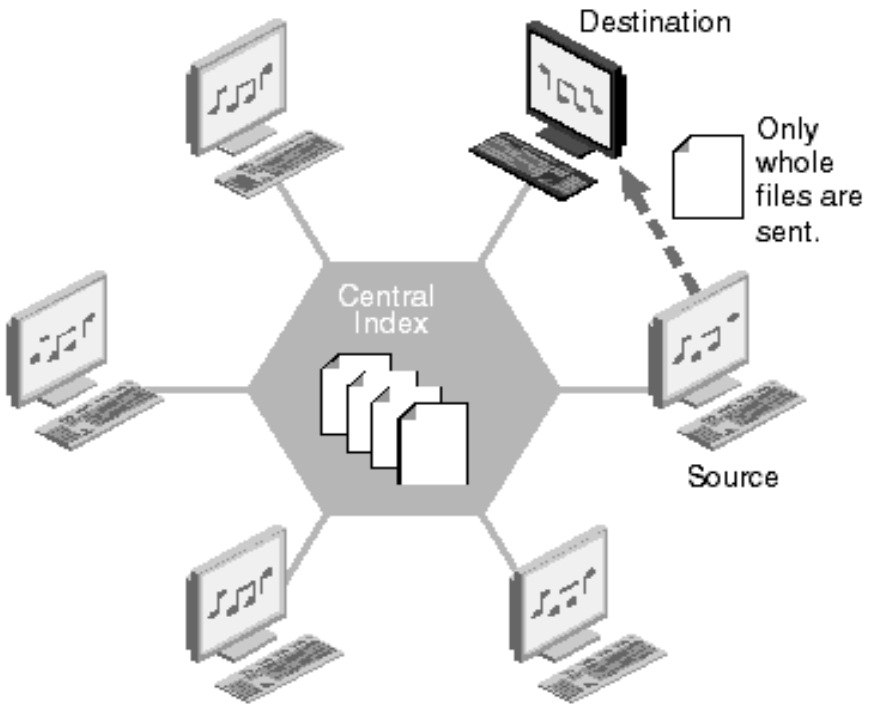

Figure 2.4: Architecture of Napster [6]

B. A Distributed Approach: Gnutella

The Gnutella protocol allows purely distributed system for file searching and sharing mechanism. Any peer in the network sends the 'ping' message to find the legitimate peers in the network. Peers receiving this ping message then broadcast with decremented Time-To-Live (TTL) and reply with the 'pong' message in order to show their presence in the network [5]. The requesting peer broadcasts a query message and it is forwarded to all the neighbouring peers. If these requested peers contain the required information, they respond back with the Query-hit message to requesting peer or they just forward the query to their neighbours as well in the case

when no information is found. This process of forwarding query is limited to a predetermined number of hops. The peer with information then has to communicate with the requesting peer along the same route as the query came from.

The main disadvantage of Gnutella is that with the increase in the number of hops for query, the response increases as logarithmic function. This has a serious issue in scalability of the network. To overcome this situation, the concept of TTL (Time-to-Live) is employed which defines the time during which query becomes active and after that, it no longer works. This has reduced the bandwidth consumption somehow.

The later version of Gnutella called Gnutella protocol version 0.6, as described in [10], have got some improvements like the ping caching which reduces the number of messages transmitted during ping-pong by implementing multiplexing, de-multiplexing of the messages. Other improvements include the implementation of two layer architecture defining a more powerful (in terms of processing capability, storage and so on) peers as the Ultra-peer and use them as proxy for low powered normal peers, so that the normal peers are shielded from excessive traffic to conserve the bandwidth.

C. A Hybrid Approach: Kazaa

Kazaa reorganizes peer nodes into a two-level hierarchy with super nodes and leaves. Super nodes are capable and reliable peer nodes that take more responsibility for providing services in the network. A super node is a temporary index server for other peer nodes. The peer nodes with high computer power and fast network connection automatically become super nodes and example is shown in *figure 2.5 [11]*.

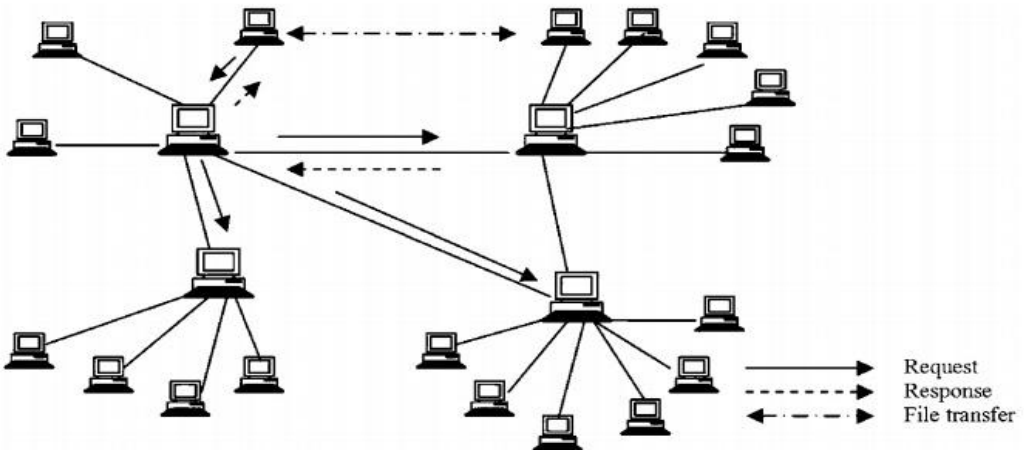

Figure 2.5: Example of Kazaa Network [11]

2.3.2. Structured P2P Overlay Network

In this overlay, the peers in network are allowed to join in a particular fashion following some scalable and routing algorithm. This helps to keep the topology somewhat systematic but one has to use great strength to maintain the structure fixed. This kind of networks is capable of implementing distributed hash table (DHT). Each node in the network is identified with a unique identifier and this ID is responded for every queries made by requesting peers in the network. This is not suitable for the peers that join and leave frequently because joining and leaving process of nodes change the structure of the network and to maintain the topology of the network, huge network resources have to be spent. However, for the network which has the peers almost static and do not join or leave the network frequently, this method sounds better, as the structure of the network can be maintained relatively longer. Several structured P2P overlay like Chord, Pastry, CAN, ROME, Accordion, Kademlia and so on have been proposed.

2.4. Machine Learning

Machine learning investigates how computers can learn or improve their performance based on data. It is the main research area for computer programs to automatically learn to recognize complex patterns and make intelligent decision based on the data. For example: a system that can automatically recognize hand written postal codes on mail after learning from a set of examples.

Machine Learning is a scientific field addressing the question "How can we program systems to automatically learn and to improve with experience?" learning is done from many kinds of experience, such as learning to predict which medical patients will respond to which treatments, by analyzing experience captured in databases of online medical records. Mobile robots learn how to successfully navigate based on experience they gather from sensors as they roam their environment, and computer aids for scientific discovery that combine initial scientific hypotheses with new experimental data to automatically produce refined scientific hypotheses that better fit observed data.

To tackle these problems, many researchers developed algorithms that discover general conjectures and knowledge from specific data and experience, based on sound statistical and computational principles. Researchers also developed theories of learning processes that characterize the fundamental nature of the computations and experience sufficient for successful learning in machines and in humans. Machine learning are of two parts i.e. supervised learning and unsupervised learning.

2.4.1. Supervised Learning

Supervised learning is fairly common in classification because goal is often to get the system to learn a classification system that we have created. Digit recognition, once again, is a common example of classification learning. Generally, classification learning is appropriate for any problem where deducing a classification is useful and the classification is easy to determine. Supervised learning is the most common technique for training neural networks and decision trees. Both of these techniques are highly dependent on the information given by the pre-determined classifications [12].

Supervised learning is the most common technique for training neural networks and decision trees. Both of these techniques are highly dependent on the information given by the pre-determined classifications [12]. Classification learning is powerful when the classifications are known to be correct (for instance, when dealing with diseases, it's generally straight-forward to determine the design after the fact by an autopsy), or when the classifications are simply arbitrary things that we would like the

computer to be able to recognize for us. Classification learning is often necessary when the decisions made by the algorithm will be required as input somewhere else. Otherwise, it wouldn't be easy for whoever requires that input to figure out what it means.

2.4.2. Unsupervised Learning

Unsupervised learning seems much harder: the goal is to have the computer learn how to do something that we don't tell it how to do! There are actually two approaches to unsupervised learning. The first approach is to teach the agent not by giving explicit categorizations, but by using some sort of reward system to indicate success. Note that this type of training will generally fit into the decision problem framework because the goal is not to produce a classification but to make decisions that maximize rewards. This approach nicely generalizes to the real world, where agents might be rewarded for doing certain actions and punished for doing others [12].

A second approach is called clustering. In this type of learning, the goal is not to maximize a utility function, but simply to find similarities in the training data. The assumption is often that the clusters discovered will match reasonably well with an intuitive classification. For instance, clustering individuals based on demographics might result in a clustering of the wealthy in one group and the poor in another [12].

Although the algorithm won't have names to assign to these clusters, it can produce them and then use those clusters to assign new examples into one or the other of the clusters. This is a data-driven approach that can work well when there is sufficient data; for instance, social information filtering algorithms, such as those that Amazon.com use to recommend books, are based on the principle of finding similar groups of people and then assigning new users to groups. In some cases, such as with social information filtering, the information about other members of a cluster (such as what books they read) can be sufficient for the algorithm to produce meaningful results. In other cases, it may be the case that the clusters are merely a useful tool for a human analyst. Unfortunately, even unsupervised learning suffers from the problem of overfitting the training data.

2.5. Classification

Classification is a data mining technique used to predict the category of categorical data by building a model based on some predictor variables (to classify data). Predictor variable/attribute is called class label attribute (predefined class). Since the training data are accompanied by labels indicating the class of the observations it is also called supervised learning and new data is classified based on the training set.

In the classification problem, the goal of the learning algorithm is to minimize the error with respect to the given inputs. These inputs, often called the "training set", are the examples from which the agent tries to learn. But learning the training set well is not necessarily the best thing to do. For instance, if I tried to teach exclusive-or, but only showed combinations consisting of one true and one false, but never both false or both true, learner might learn the rule that the answer is always true.

It is a two-step process

1. Model Construction (Learning step or Training Phase)
   - Build a model to explain the target concept
   - model is represented as classification rules, decision trees, or mathematical formulae
2. Model Usage (Testing Phase)
   - is used for classifying future or unknown cases
   - estimate the accuracy of the model

2.5.1. Decision Tree

A decision tree is a flowchart-like tree structure, where each internal node (non-leaf node) denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (or terminal node) holds a class label. The topmost node in a tree is the root node. A typical decision tree is shown in *figure 2.7* [13].

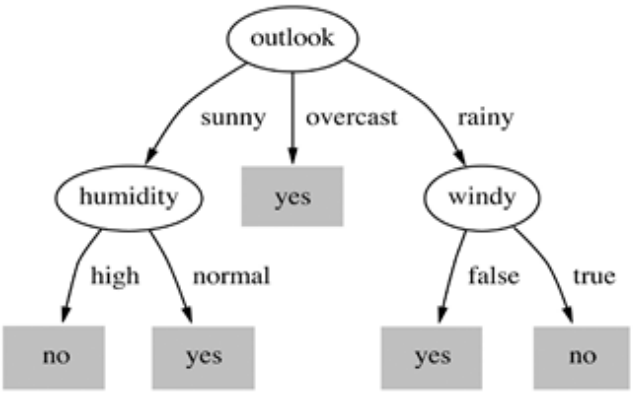

Figure 2.6: Decision tree example [13]

During the late 1970s and early 1980s J. Ross Quinlan, a researcher in machine learning, developed a decision tree algorithm known as **ID3** (Iterative Dichotomiser). This work expanded on earlier work on concept learning system, described by E. B Hunt, J. Marin, and P. T. Stone. Quinlan later presented C4.5 (a successor of ID3), which become a benchmark to which newer supervised learning algorithms are often compared. In 1984, a group of statisticians published the book classification and regression trees (CART), which described the generation of binary decision trees.

- **Decision Tree Construction Algorithm**

Input: A data set, D

Output: A decision tree

- If all the instances have the same value for the target attribute then return a decision tree that is simply this value (not really a tree - more of a stump).
- Else

1. Compute Gain values for all attributes and select an attribute with the highest value and create a node for that attribute.
2. Make a branch from this node for every value of the attribute
3. Assign all possible values of the attribute to branches.
4. Follow each branch by partitioning the dataset to be only instances whereby the value of the branch is present and then go back to 1.

- **Attribute Selection Measures**

In a data set there are lots of attributes and we do have problem on selection of attribute as node and as leaf. There arise questions **which attribute first?**

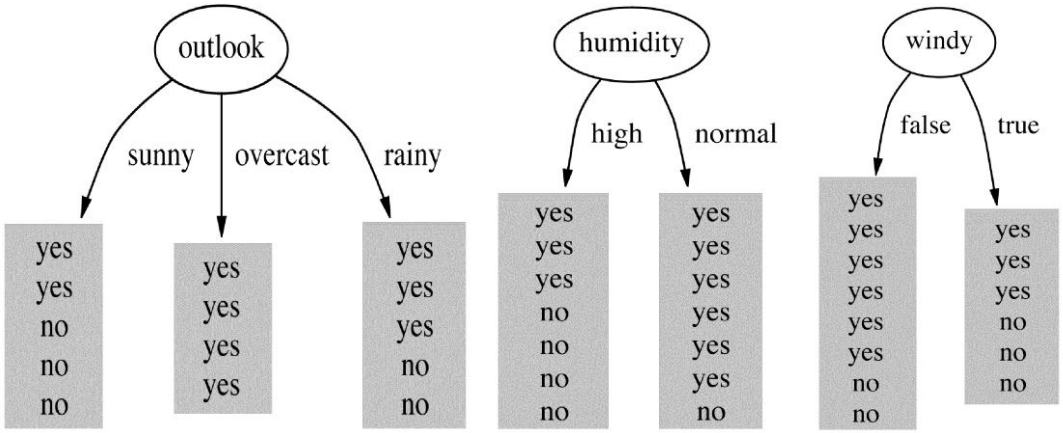

Figure 2.7: Possibility of attribute as node [13]

Attribute selection measure [13] is a heuristic for selecting the splitting criterion that "best" separates given data partition, D, of class-labeled training tuples into individual classes. Attribute selection measures are also known as splitting rules because they determine how the tuples at a given node are to be split. The attribute selection measure provides a ranking for each attribute describing the given training tuples. The attribute having the best score for the measure is chosen as the splitting attribute for the given tuples.

- **Information Gain**

ID3 uses information gain as its attribute selection measure. This measure is based on pioneering work by Claude Shannon on information theory, which studied the value or "information content" of messages [13].

Information gain = (information before split) – (information after split) bits

$$Gain(A) = Info(D) - Info_A(D)\ bits$$ ----------Equation 2.1

Where,

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)\ bits$$ ----------Equation 2.2

- P_i = $|Ci,D| / |D|$
- A having v distinct values, $\{a_1, a_2, \ldots, a_v\}$
- $D_1, D_2, \ldots, D_v$ then,

$$Info_A(D) = \sum_{j=1}^{v} \frac{|Dj|}{|D|} \times Info(Dj)\, bits$$ ----------Equation 2.3

- **Gain Ratio:**

The information gain [13] measure is biased toward tests with many outcomes. That is, it prefers to select attributes having a large number of values. C4.5, a successor of ID3, uses an extension to information gain known as **gain ratio**, which attempts a overcome this bias. It applies a kind of normalization to information gain using a "Split information" value define analogously with $Info(D)$ as

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|Dj|}{|D|} \times \log_2\left(\frac{|Dj|}{|D|}\right) bits$$ ----------Equation 2.4

The gain ratio is defined as

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo_A(D)}$$ ----------Equation 2.5

The attribute with the maximum gain ratio is selected as the splitting attribute. Note, however, that are the split information approaches 0, the ratio becomes unstable. A constraint is added to avoid this, whereby the information gain of the test selected must be large-at last as great as the average gain over all tests examined.

- **Gini Index**

The Gini index [13] is used in CART. Using the notation previously described, the Gini index measures the impurity of D, a data partition or set of training tuples, as

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2\, bits$$ ----------Equation 2.6

Where, P_i is the probability that a tuple in D belong to class C_i and is estimated by $|C_{i,D}| / |D|$. The sum is computed over m classes. The Gini index considers a binary

split for each attribute. Let's first consider the case where A is a discrete-valued attribute having v distinct values, {a1, a2, ..., av}, occurring in D. If A has v possible values, then there are 2^v possible subsets but we exclude the power set, and the empty set from consideration since, conceptually, they do not represent a split. Therefore there are 2^v-2 possible ways to form two partitions of the data, D, based on a binary split on A.

When considering split, we compute a weighted sum of the impurity of each resulting partition. For example, if a binary split on A partitions D into D_1 and D_2, the Gini index of D given that partitioning is

$$Gini_A(D) = \left\{ \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2) \right\} bits$$ ----------Equation 2.7

For each attribute, each of these possible binary splits is considered. For discrete-valued attribute, the subset that gives the minimum Gini index for that attribute is selected as its splitting subset.

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

$$\Delta Gini(A) = \{Gini(D) - Gini_A(D)\} bits$$ ----------Equation 2.8

The attribute that maximizes the reduction in impurity (or, equivalently, has the minimum Gini index) is selected as the splitting attribute.

2.5.2. Naive-Bayes

The naïve Bayesian [13] classifier, or simple Bayesian classifier, works as follows:

1 Let *D* be a training set of tuples and their associated class labels. As usual, each tuple is represented by an *n*-dimensional attribute vector, $\boldsymbol{X} = (x_1, x_2, \ldots, x_n)$, depicting *n* measurements made on the tuple from *n* attributes, respectively, $A_1, A_2, \ldots, A_n$.

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

$$P(C_i|\boldsymbol{X}) > P(C_j|\boldsymbol{X}) \text{ for } 1 \_ j \_ m, j \neq i.$$

Thus we maximize $P(C_i|\boldsymbol{X})$. The class C_i for which $P(C_i|\boldsymbol{X})$ is maximized is called the *maximum posteriori hypothesis*. By Bayes' theorem,

$$P(C_i \mid X) = \frac{P(X \mid C_i)P(C_i)}{P(X)} \quad \text{----------Equation 2.9}$$

3 As $P(\boldsymbol{X})$ is constant for all classes, only $P(\boldsymbol{X}|C_i)$ $P(Ci)$ need be maximized. If the class prior probabilities are not known, then it is commonly assumed that the classes are equally likely, that is, $P(C_1) = P(C_2) = \ldots\ldots = P(C_m)$, and we would therefore maximize $P(\boldsymbol{X}|C_i)$. Otherwise, we maximize $P(\boldsymbol{X}|C_i)$ $P(C_i)$. Note that the class prior probabilities may be estimated by $P(C_i)=|C_i,D|/|D|$,where $|C_i,D|$ is the number of training tuples of class C_i in D.

4 Given data sets with many attributes, it would be extremely computationally expensive to compute $P(\boldsymbol{X}|C_i)$. In order to reduce computation in evaluating $P(\boldsymbol{X}|C_i)$, the naive assumption of class conditional independence is made. This presumes that the values of the attributes are conditionally independent of one another, given the class label of the tuple (i.e., that there are no dependence relationships among the attributes). Thus,

$$\begin{aligned} P(X \mid C_i) &= \prod_{k=1}^{n} P(x_k \mid C_i) \\ &= P(x_1 \mid C_i) \times P(x_2 \mid C_i) \times P(x_3 \mid C_i) \ldots\ldots\ldots \times P(x_n \mid C_i) \end{aligned} \quad \text{---Equation 2.10}$$

2.5.3. NBTree

The NBTree [14] algorithm is similar to the classical recursive partitioning schemes, except that the leaf nodes created are NB categorizers instead of nodes predicting a single class. Kohavi [15] proposes to deploy a naïve Bayes in each leaf, and the resulting decision tree is called an NBTree. The algorithm for learning an NBTree is similar to C4.5. After a tree is grown, a naïve Bayes is constructed for each leaf using

the data associated with that leaf. An NBTree classifies an example by sorting it to a leaf and applying the naïve Bayes in that leaf to assign a class label to it.

2.6. Application Programing Interface (API) for Data Mining

Application Programming Interface (API) [16] is a set of routines used by an application program to direct the performance of procedures by the computer's operating system. To achieve our goal we will be using most commonly used Machine Learning API.

2.6.1. WEKA

Weka [2, 17, 18] is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes [19].

Weka was developed at the University of Waikato in New Zealand; the name stands for *Waikato Environment for Knowledge Analysis*. (Outside the university, the Weka, pronounced to rhyme with *Mecca*, is a flightless bird with an inquisitive nature found only on the islands of New Zeland.) The system is written in Java and distributed under the terms of the General Public License (GNU). It runs in almost any platform and has been tested under Linux, Windows and Macintosh operating systems- and even on a personal digital assistant [19]. Weka's native data storage method is *Attribute-Relation File Format* (ARFF) [20].

2.7. Related Works

Research in P2P systems, such as Chord [21], CAN [22], Pastry [23], P2P-Net [24] or P-Grid [25] is based on various forms of Distributed Hash Tables (DHTs) and supports mappings from keys, e.g., titles or authors, to locations in a decentralized manner such that routing scales well with the number of peers in the system.

Typically, an exact-match key lookup can be routed to the proper peer(s) in at most O(log n) hops, and no peer needs to maintain more than O(log n) routing information. These architectures can also cope well with failures and the high dynamics of a P2P system as peers join or leave the system at a high rate and in an unpredictable manner. However, the approaches are limited to exact-match, single keyword queries on keys. This is insufficient when queries should return a ranked result list of the most relevant approximate matches [26].

In the following we briefly discuss some existing approaches towards P2P web search. Galanx [27] is a peer-to-peer search engine implemented using the Apache HTTP server and BerkelyDB. It directs user queries to relevant nodes by consulting local peer indexes similar to our approach.

A. Ismail et al. [2, 17] and C. Indra [1] had proposed an unstructured P2P system based on an organization of peers around Super`-Peers according to their semantic domains, where Super-Peers are connected to a Super-Super-Peer that is the engine that specifies the Super-Peer that has the peers which may have relevant data to answer to queries. In order to query A. Ismail et al. had used J48 algorithm, which is Weka's implementation of C4.5 Decision tree algorithm and C. Indra [1] had used GATreee but still there is challenge how to locate peers that are relevant with respect to a given query with minimum query processing and answering time.

S. Datta, K. Bhaduri, C. Giannella, R. Wolff, H. Kargupta [28] Said Distributed data mining is a natural choice when the data mining environment has distributed data, computing resources, and users. They focused on an emerging branch of distributed data mining—peer-to-peer data mining. P2P data mining applications may play a key role in the next generation of file sharing networks, sensor networks, and mobile ad hoc networks. Their paper also offered a sampler of exact and approximate P2P algorithms for clustering in such distributed environments.

F. Dewan Md and R. Mohammad Zahidur [29] presents in therir research; a hybrid approach to intrusion detection based on decision tree-based attribute weighting with naïve Bayesian tree, which is suitable for analyzing large number of network logs. The main propose of this paper is to improve the performance of naïve Bayesian classifier for network intrusion detection systems (NIDS). The experimental results manifest that proposed approach can achieve high accuracy in both detection rates and false positives, as well as balanced detection performance on all four types of network intrusions in KDD99 dataset.

In paper [30], researchers developed a qualitative information security risk assessment methodology by the aid of machine learning classifier algorithms (BayesNet, Lazy. LBR, NB Tree, VFI and DTNB) that was successfully implemented in the human resources department of a logistics company.

H. Ahmed et al. [31] had shown that NBTree had classified bettern among his compared classification algorithms. S. Bekti Maryuni [32] had also show that the application of a hybrid approach to naïve Bayes decision tree can improve the performance of naïve Bayes, although still below the decision tree.

CHAPTER 3

RESEARCH METHODOLOGY

3.1. Background

This chapter deals with the framework and algorithm of the proposed P2P system architecture model.

3.2. P2P System Architecture Model

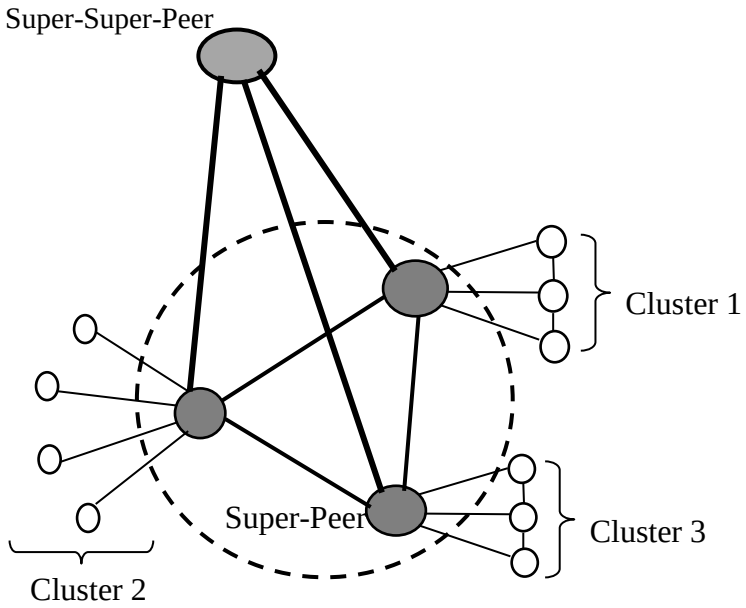

Figure 3.1: Semantic Network of Super-Peer organized by cluster with Super-Super-Peer [1, 2, 17].

Our proposed system; Figure 3.1 [1, 2, 17] was an unstructured P2P system based on an organization of peers around Super-Peers according to their semantic domains, where Super-Peers are connected to a Super-Super-Peer which specifies the Super-Peer that have Peers which may have relevant data to answer the given queries. This architecture combines centralized approach and unstructured taking the advantages of centralized research and autonomy, the distribution of loads and robustness for a distributed search. The Super-Peers architecture allows for the heterogeneity of Peers by assigning more responsibility to Peers able to assume them. Therefore Peer called Super-Super-Peer has

- An additional computing power
- Greater bandwidth
- Performs administrative task
- Manages all Super-Peers
- Reduce efforts of compilation of Queries
- Have global index using decision tree to identify relevant Super-Peers

3.3. Algorithms

In this research, two algorithms are proposed to implement i.e. one is algorithm for queries routing and another is NBTree: The hybrid algorithm.

3.3.1. Query Routing Algorithm

Assuming that P1 issues a query Q, the query routing algorithm [1, 2, 17] proceeds as follows:

- We first find the responsible Super-Peer for P1 which in this example is Super-Peer1
- The responsible Super-Peer sends the query to the Super-Super-Peer to identify the relevant Super-Peers for this query
- The Super-Super-Peer will send the query to all relevant Super Peers
- Each relevant Super-Peer treats query to find relevant peers
- Then the final set of relevant peers and their corresponding Super-Peers are returned.

3.3.2. NBTree: The Hybrid Algorithm [15]

Input: a set T of labelled instances.

Output: a decision-tree with Naive-bayes categories at the leaves.

1. For each attribute X_i, evaluate the utility, $u(X_i)$, of a split on attribute X_i, For continous attributes, a threshold is also found at this stage.
2. Let $j = \arg\max_i (u_i)$, i.e., the attribute with the highest utility.
3. If u_j is not significantly better than the utility of the current node, create a Naive-Bayes classifier for the current node and return.
4. Partition T according to the test on X_j, If X, is continuous, a threshold split is used; if X, is discrete, a multi-way split is made for all possible calues.
5. For each child, call the algorithm recursively on the portion of T that matches the test leading to the child.

   Given *m* instances, *n* attributes and *l* label values, the complexity of the attribute selection phase for discretized attributes is $O(m.\ n^2.\ l)$. If the number of attributes is less than $O(\log m)$ and the number of labels is small, then the time spent on attribute selection using cross-validation is less than the time spent sorting the instances by each attribute.

3.4. Source of Data

Datasets were from secondary source of data.

3.5. Experiments Protocol

Experiment were done by using machine learning tool Weka V 3.7.10.

3.5.1. Experimental Setup and Evaluation

Table 3.1: NBTree Experimental Parameters

Scheme : weka.classifiers.trees.NBTree
File Format : arff
Instances : about 5,001 to 45,000
Test mode : 10-fold cross-validation

CHAPTER 4

EXPERIMENT AND RESULT ANALYSIS

4.1. Background

This section deals with the successful implementation and *Analysis of Queries Routing in Super-Super-Peer based P2P architecture using NBTree: the hybrid algorithm*. The experiments were performed in famous data mining tool Weka installed in the system consist of Intel® core™ i3-3110M CPU @ 2.40GHz with 4.00 GB RAM in 32 bit Windows 7 Professional Operating System.

4.2. Tool

The system can be implemented using data mining tools. In this research, Weka V3.7.10 was used.

Weka [2, 17, 18] is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes [19].

Weka was developed at the University of Waikato in New Zealand; the name stands for *Waikato Environment for Knowledge Analysis*. (Outside the university, the Weka, pronounced to rhyme with *Mecca*, is a flightless bird with an inquisitive nature found only on the islands of New Zeland.) The system is written in Java and distributed under the terms of the General Public License (GNU). It runs in almost any platform and has been tested under Linux, Windows and Macintosh operating systems- and even on a personal digital assistant [19]. Weka's native data storage method is *Attribute-Relation File Format* (ARFF) [20].

4.3. Data Structure

The main data structures used in this study are enlisted below:

- Randomly generated data sets in the scenario of real world data were used.
- Datasets were consist of 7 attributes; Super-Peers, Queries and Peers, Keywords (ComponentW1, ComponentW2, ComponentW3 and ComponentW4).
- Used datasets were in *Attribute-Relation File Format* (ARFF).
- Used datasets consists of 300 Peers, 10 Super-Peers and ranged from 5,001 instances to 45,000 instances.

4.4. Data Samples

Sample of data set used in the system is shown below.

ARFF-Viewer - D:\Thesis work\P2P Dataset\Datasets for Experiment\CSV\P2P dataset 1.csv

File Edit View

P2P dataset 1.csv

Relation: P2P dataset 1

No.	1: Peer Nominal	2: Query Nominal	3: ComponentW1 Nominal	4: ComponentW2 Nominal	5: ComponentW3 Nominal	6: ComponentW4 Nominal	7: **Super Peer** Nominal
1	P225	Q226	i.c	r.m	r.k	g.f	SP0
2	P53	Q58	d.m	i.c	h.i	d.o	SP1
3	P126	Q268	d.o	r.m	g.f	i.c	SP2
4	P58	Q53	f.e	r.m	h.s	i.c	SP3
5	P274	Q225	i.c	r.k	r.m	k.f	SP4
6	P191	Q241	d.o	i.c	g.f	d.m	SP5
7	P110	Q286	g.f	f.e	k.f	k.l	SP6
8	P131	Q153	i.c	d.o	r.k	d.m	SP7
9	P273	Q23	g.f	r.k	f.e	h.i	SP8
10	P199	Q260	r.k	f.e	i.c	k.f	SP9
11	P126	Q268	d.o	r.m	g.f	i.c	SP2
12	P274	Q225	i.c	r.k	r.m	k.f	SP6
13	P236	Q23	g.f	r.k	f.e	h.i	SP8
14	P110	Q286	g.f	f.e	k.f	k.l	SP6
15	P306	Q124	r.k	d.m	r.m	g.f	SP3
16	P48	Q241	d.o	i.c	g.f	d.m	SP3
17	P131	Q199	k.f	g.f	i.c	d.o	SP0
18	P199	Q260	r.k	f.e	i.c	k.f	SP9
19	P131	Q199	k.f	g.f	i.c	d.o	SP0
20	P131	Q153	i.c	d.o	r.k	d.m	SP7
21	P225	Q226	i.c	r.m	r.k	g.f	SP0
22	P199	Q58	d.m	i.c	h.i	d.o	SP0
23	P241	Q158	d.o	k.l	r.k	r.m	SP7
24	P131	Q199	k.f	g.f	i.c	d.o	SP0
25	P48	Q241	d.o	i.c	g.f	d.m	SP5
26	P126	Q268	d.o	r.m	g.f	i.c	SP2
27	P199	Q260	r.k	f.e	i.c	k.f	SP9
28	P225	Q23	g.f	r.k	f.e	h.i	SP8

Figure 4.1: Portion of input samples viewed in WEKA ARFF-Viewer

4.5. Experiments and Results

Evaluating the performance of P2P network is an important part to understand how useful it can be in the real world. As with all P2P applications, the first question is whether P2P is scalable. Our system was evaluated with NBTree: The Hybrid Algorithms and with different set of parameters (i.e. datasets consists of different instances). Evaluated results were quite encouraging.

4.5.1. Experiment

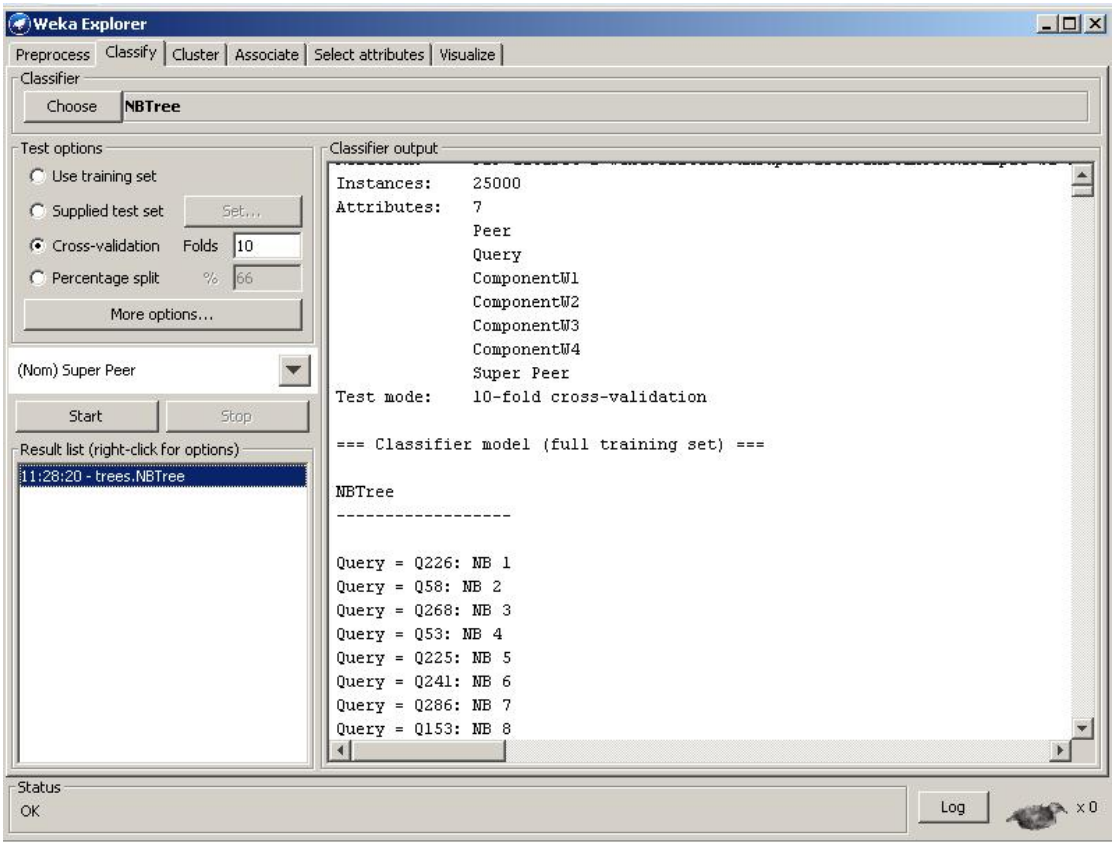

Figure 4.2: Result of running NBTree: The Hybrid Algorithm

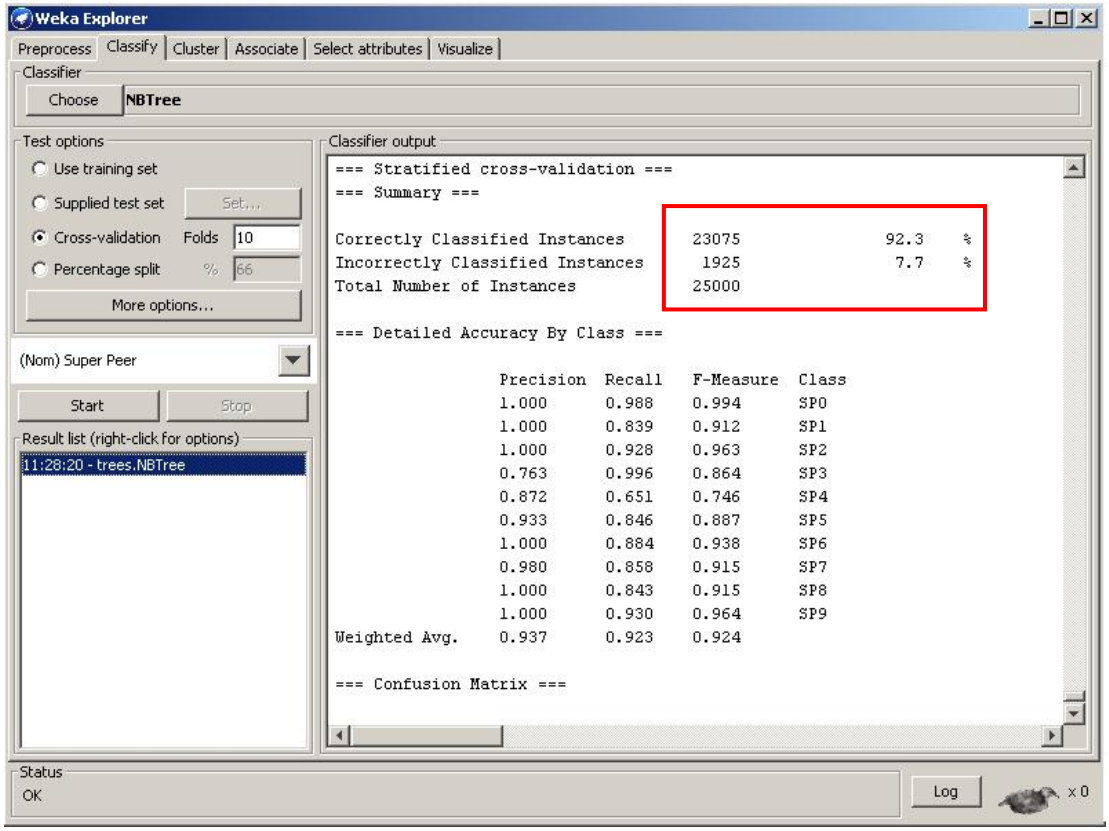

Figure 4.3: Result of classified instances by using NBTree

The output from the WEKA program is shown in the Figure 4.3. In this output, the decision tree by NBTree was able to classify 92.30 % of the data correctly.

4.5.2. Evaluation

For the evaluation followings are evaluation metrics.

- **10-fold Cross Validation**

In **10-fold cross-validation**, the initial data are randomly partitioned into 10 mutually exclusive subsets or "folds" i.e. D_1, D_2, D_3, …… , D_{10} each of approximately equal size. Training and testing is performed 10 times in the ratio of 9:1 means to say 9 fold as Training and 1 fold as Testing.

- **Confusion Matrix**

A confusion matrix is a table for analyzing the result of the classifiers. It deals with how classifier can recognize tuples of different classes. In order to develop the confusion matrix, the following terms should be considered:

- **True Positive (TP):** Positive tuples that are correctively labelled by the classifier.
- **True Negative (TN):** Negative tuples that are correctly labelled by the classifier.
- **False Positive (FP):** Negative tuples that are incorrectly labelled as positive.
- **False Negative (FN):** Positive tuples that are mislabelled as negative.

Table 4.1: Confusion Matrix

		Predicted Class		
		Yes	No	Total
Actual Class	Yes	**TP**	**FN**	**P**
	No	**FP**	**TN**	**N**
	Total	**P′**	**N ′**	**P+N**

- **Accuracy**

Accuracy of a classifiers on a given test set is the percentage of test set tuples that are correctly classified by the classifiers. It also refers to the recognition rate of the classifier that means how the classifier recognizes tuples of the various classes.

$$\text{Accuracy} = \frac{TP + TN}{P + N}$$ ---------- Equation 4.1

- **Precision**

Precision refers to the measure of exactness that means what percentage of tuples labeled as positive are actually such.

$$\text{Precision} = \frac{TP}{TP + FP}$$ ----------Equation 4.2

- **Recall**

Recall refers to the true positive rate that means the proportion of positive tuples that are correctly identified. It is also known as sensitivity of the classifier.

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{TP}{P}$$ ----------Equation 4.3

- **F-Measure**

The F-score or F-Measure also refers to F-measures combines the both the measures Precision and Recall as the harmonic mean

$$\text{F} - \text{Measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precesion} + \text{Recall}}$$ ----------Equation 4.4

- **Time Taken to Build Model (TTBM)**

The time taken to build model refers to total time taken by a fold of the simulation tool to build a model to train the machine. Time is calculated in second.

4.5.3. Results

Table 4.2: Result of NBTree

Relation	Total Instances	Accuracy (%)	Precision (%)	Recall (%)	F-Measure (%)	Tree Size	Time Taken to Build Model (Sec)
D1	5,001	91.86	93.5	91.9	92	74	2.62
D2	10,000	92.16	93.5	92.2	92.3	132	2.17
D3	15,000	92.10	93.5	92.1	92.2	105	2.76
D4	20,000	92.27	93.6	92.3	92.4	51	3.36
D5	25,000	92.30	93.7	92.3	92.4	51	4.04
D6	30,000	92.20	93.7	92.2	92.3	51	4.36
D7	35,000	92.19	93.6	92.2	92.3	51	5.15
D8	40,000	92.16	93.6	92.2	92.3	24	5.59
D9	45,000	92.21	93.7	92.2	92.4	24	6.32

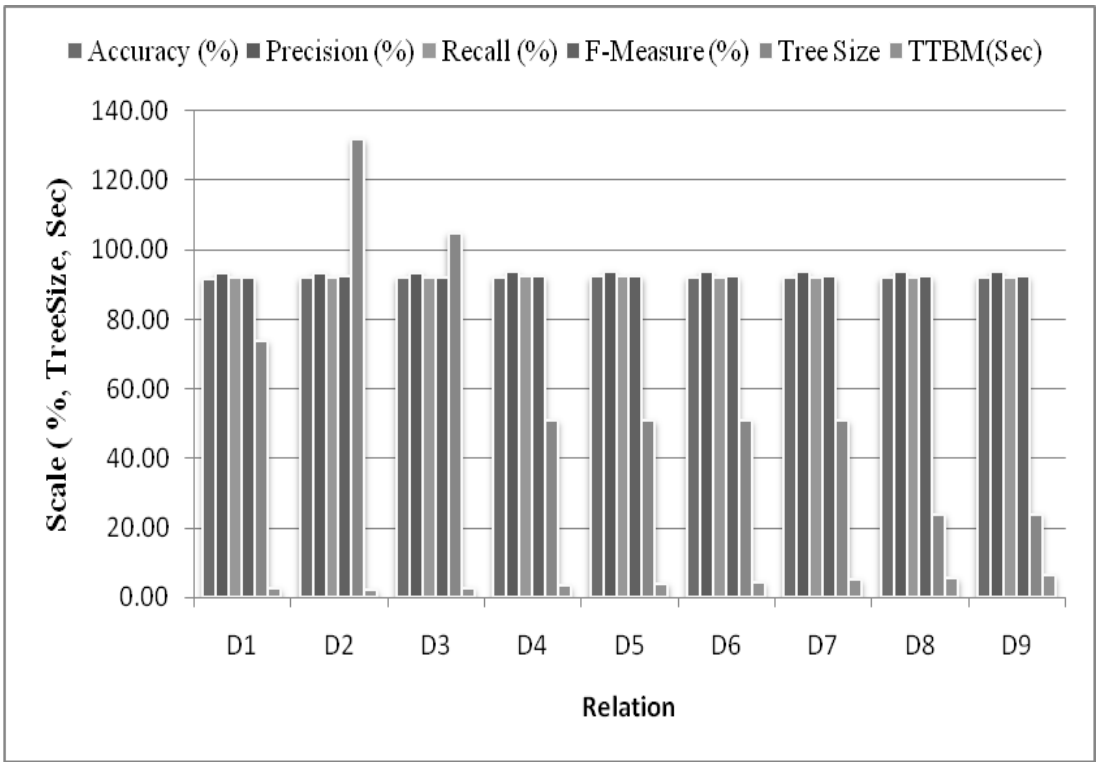

Figure 4.4: Graph of Table 4.2

Table 4.3: Result of NBTree taking Accuracy

Relation	**Total Instances**	**Accuracy (%)**
D1	5,001	91.86
D2	10,000	92.16
D3	15,000	92.10
D4	20,000	92.27
D5	25,000	92.30
D6	30,000	92.20
D7	35,000	92.19
D8	40,000	92.16
D9	45,000	92.21

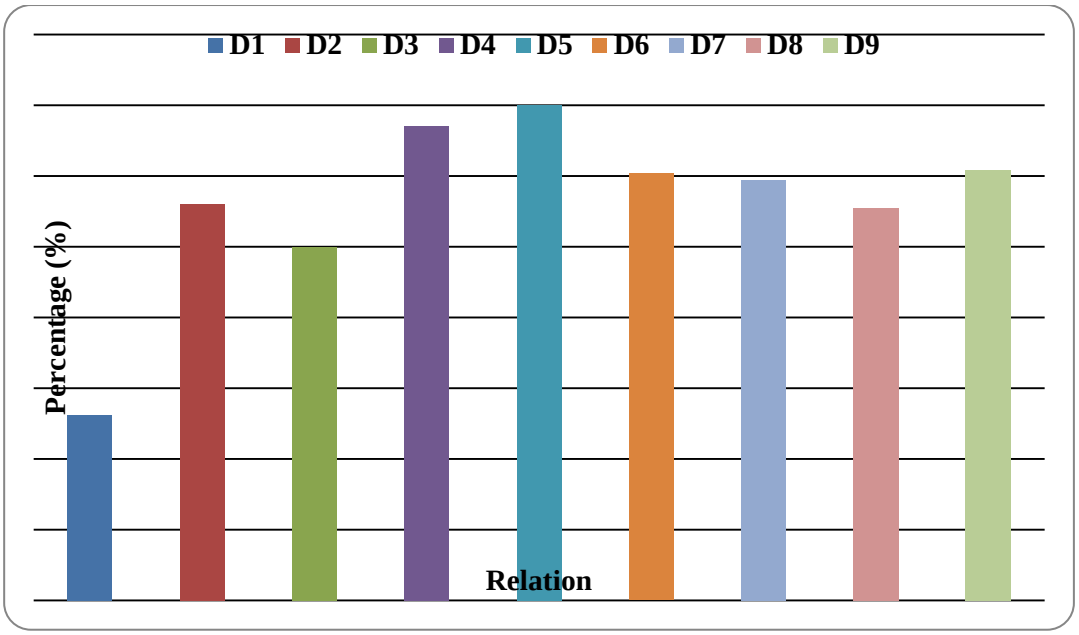

Figure 4.5: Graph of Table 4.3

Table 4.4: Result of NBTree taking Precision, Recall and F-Measure

Relation	Total Instances	Precision (%)	Recall (%)	F-Measure (%)
D1	5,001	93.5	91.9	92
D2	10,000	93.5	92.2	92.3
D3	15,000	93.5	92.1	92.2
D4	20,000	93.6	92.3	92.4
D5	25,000	93.7	92.3	92.4
D6	30,000	93.7	92.2	92.3
D7	35,000	93.6	92.2	92.3
D8	40,000	93.6	92.2	92.3
D9	45,000	93.7	92.2	92.4

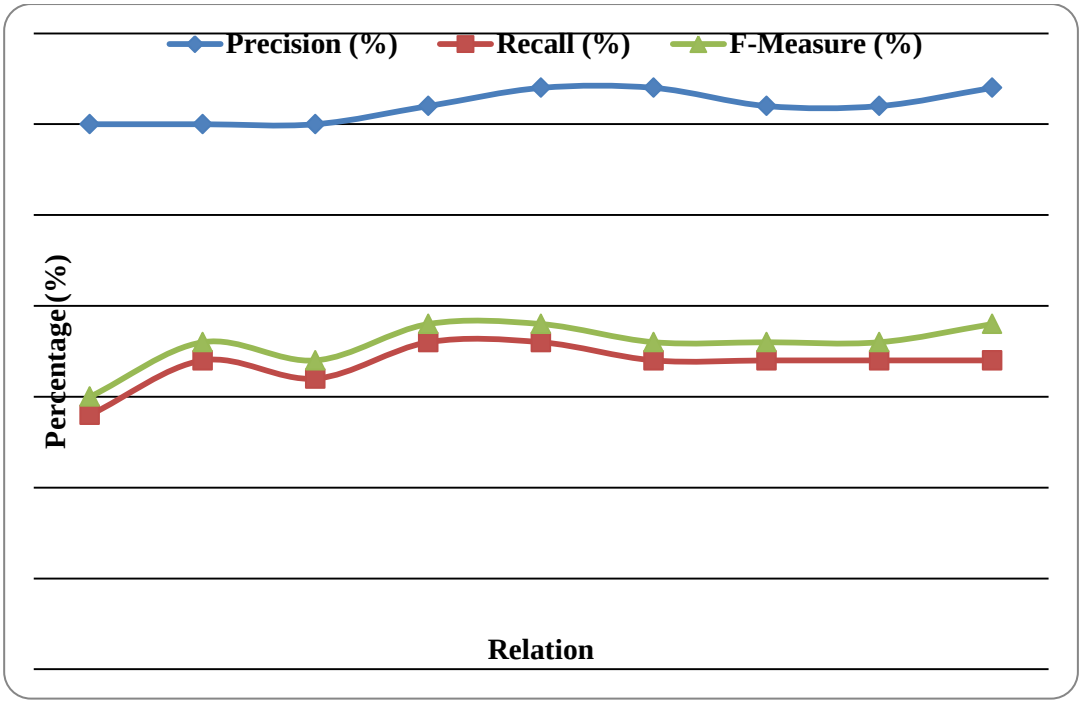

Figure 4.6: Graph of Table 4.4

Table 4.5: Result of NBTree taking Tree Size

Relation	Total Instances	Tree Size
D1	5,001	74
D2	10,000	132
D3	15,000	105
D4	20,000	51
D5	25,000	51
D6	30,000	51
D7	35,000	51
D8	40,000	24
D9	45,000	24

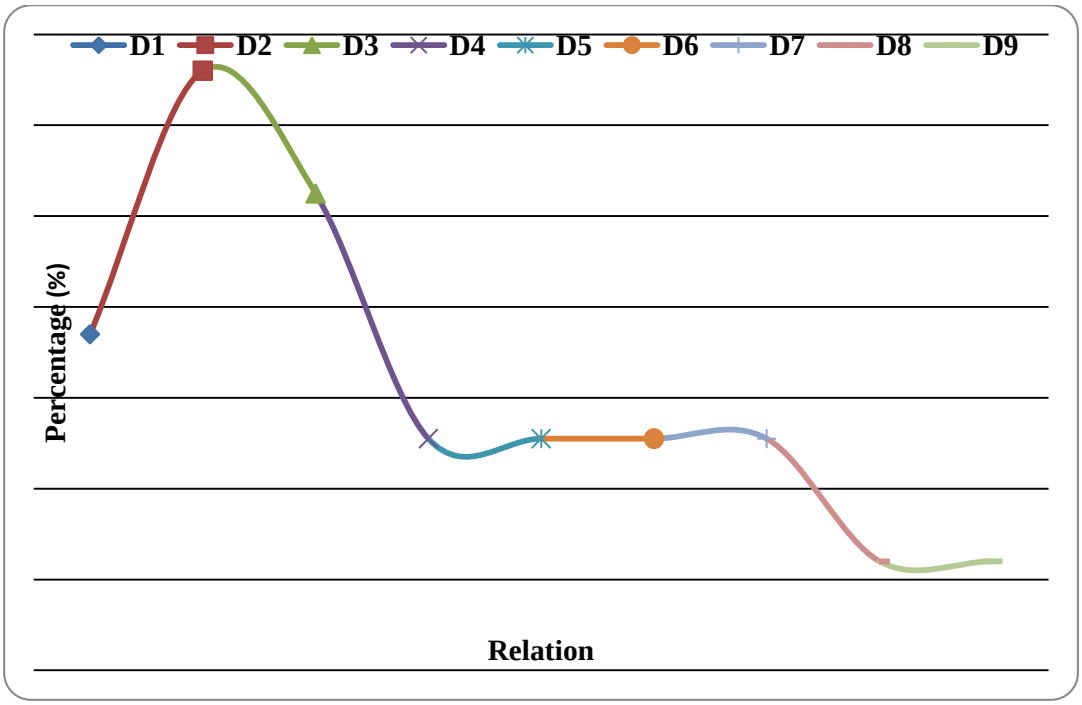

Figure 4.7: Graph of Table 4.5

Table 4.6: Result of NBTree taking Time Taken to Build Model (TTBM)

Relation	Total Instances	TTBM (Sec)
D1	5,001	2.62
D2	10,000	2.17
D3	15,000	2.76
D4	20,000	3.36
D5	25,000	4.04
D6	30,000	4.36
D7	35,000	5.15
D8	40,000	5.59
D9	45,000	6.32

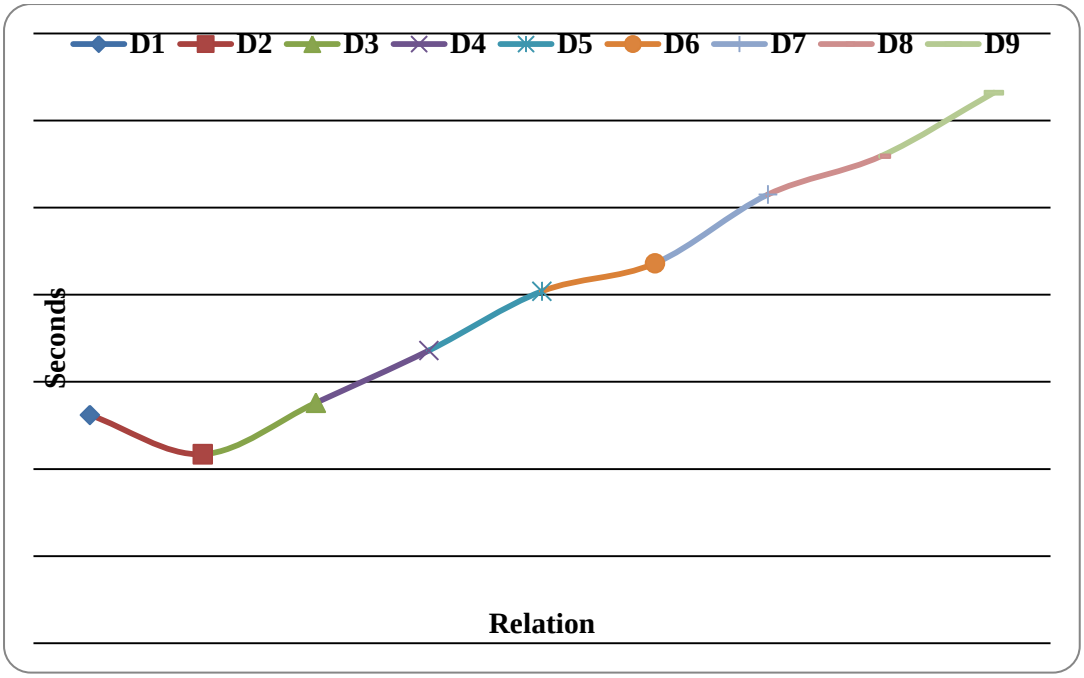

Figure 4.8: Graph of Table 4.6

4.6. Context Analysis

The table 4.2, 4.3, 4.4, 4.5 and 4.6 as well as graph shown in figures 4.4, 4.5, 4.6, 4.7, and 4.8 are the results of the simulations. They demonstrate the performance of using the Super-Super-Peer with decision tree for the discoveries of P2P communities in proposed system model.

Figure 4.4 shows the comparision between Accuracy, Precision, Recall, F-measure, Tree Size and Time Taken to Build Model whereas figure 4.5 shows that Accuracy of decision tree observed by NBTree. It showed that the classification accuracy of NBTree ranges from 91.86% to 92.30%.

In Figures 4.6 are the results of NBTree where comparision of Precison, Reccall and F-Measure are shown for the validation. It showed that values of F-measure always lies between Precision and Recall, which validates the proposed system as well as the defination of F-measure that F-measure is harmonic mean of Precision and recall.

Figure 4.7 shows the Tree Size generated by NBTree which ranges from 24 to 132 with average tree size of 62.56 and Figure 4.8 shows the results of Time Taken to Build Model (in Seconds) where NBTree had consumed time from 2.17 seconds to 6.32 seconds which shows that time consumption increases along with the increase in instances.

CHAPTER 5

CONCLUSION AND FUTURE WORKS

5.1. Conclusion

In recent years, database systems have become highly distributed. The distributed system paradigms, such as peer-to-peer databases, are being adopted. A challenging problem in unstructured P2P system is to locate relevant peers with respect to a given query with minimum query processing and minimum answering time because peers can leave the network and new peers can join it any time. This thesis work suggested an unstructured P2P system which is based on an organization of peers around Super-Peers that is connected to Super-Super-Peer according to their semantic domains.

Classification based on decision trees is one of the challenging problems in Machine Learning. This research had implemented NBTree: The Hybrid Algorithm to extract Super-Peer that contains peers with relevant data that respect to a given query. From the context analysis it is seen that NBTree has similar range accuracy as existing results given by J48 algorithm and far better than GATree algorithm in existing research papers. Where as NBTree had shown far smaller tree size than result of J48 algorithm of existing papers and little bigger than result of GATree algorithm of existing papers. NBTree had also shown that Time Taken to Build Model increases along with the increase in the instances.

Since NBTree posses same accuracy level as J48 with smaller tree size than J48 and it posses same range of tree size and far better accuracy than GATree this research proves that instead NBTree can be used for good accuracy as well as smaller tree size for the fast query processing. Also since, this research showed that same accuracy range though the instances were scaled up which proved the scalability of the proposed system model.

5.2. Future Works

Directions for future works are:

- One important area for improvement is performance (Answering time).
- Another is enhancing the performance more (Answering time) by clustering the Super-Peers into multiple Super-Super-Peer to minimize the load on one Super-Super-Peer and also to be more scalable.

REFERENCES

[1] C. Indra, “Data Mining of Queries Routing in Super-Super-Peer Based P2P Architecture using Decision Tree”, M.Sc Thesis, Dept. Grad. Stud., NCIT Pokhara Univ., Lalitpur, NP, Aug. 2013.

[2] I. Anish, Q. Mohamand, N. Gille, H. Mohamad, "Data Mining in P2P Queries Routing Using Decision Trees", *5th international Conference: Sciences of Electronic, Technologies of Information and Telecommunications*, Tunisia, March 22-26, 2009.

[3] D. S. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, Z. Xu, "Peer-to-Peer computing", Technical Report HPL-2002-57, HP Laboratories Palo Alto, 2002.

[4] B. H. Park and H. Kargupta, "Distributed Data Mining: algorithms, systems, and applications", In N. Ye, editor, *The Hand book of Data mining*, Lawrence Eribaum Associates, Inc., 2003.

[5] Erkki Harjula, Mika Ylianttila, Jussi Ala-Kurikka, Jukka Riekki, Jaakko Sauvola, “Plug-and-play application platform: towards mobile peer-to-peer”, *Proceedings of the 3rd International Conference on Mobile and Ubiquitous Multimedia-(NUM04)*, Publisher: ACM, Publication year: October 2004.

[6] http://www.napster.com (last accessed on December, 2013).

[7] M. Ripeanu, "Peer-to-Peer Architecture Case Study: Gnutella network", *Proceedings of the First International Conference on Peer-to-Peer Computing*, Pages: 99-100, Publication year: 2001.

[8] Dinesh C. Verma, "Legitimate Applications of Peer-to-Peer Networks", John Wiley & Sons, Inc., 2004.

[9] Dana Moore, John Hebeler, "Peer-to-Peer: Building Secure, Scalable, and Manageable Networks", McGraw Hill/Osborne, 2002.

[10] Xing Jin, S.-H. Gary Chan, "Unstructured Peer-to-Peer Network Architectures", X. Shen et al. (eds.), *Handbook of Peer-to-Peer Networking*, DOI 10.1007/978-0-387-09751-0_5, © Springer Science+Business Media, LLC 2010.

[11] Lu Liu, Nick Antonopoulos, "From Client-Server to P2P Networking", X. Shen et al. (eds.), *Handbook of Peer-to-Peer Networking*, DOI 10.1007/978-0-387-09751-0_3, © Springer Science+Business Media, LLC 2010.

[12] http://www.aihorizon.com/essays/generalai/supervised_unsupervised_machine_learning.htm (last accessed on January 10, 2013)

[13] H. Jawei, K. Micheline, P. Jain, "Data Mining: Concepts and Techniques, Third Edition", Morgan Kaufmann Publishers an imprint of Elsevier, 225 Wyman Street, Waltham, MA 02451, USA 2012.

[14] L. Wen-Chen and C. Bor-Wen, "An Intelligent System for Improving Performance of Blood Donation." *Journal of Quality Vol* 18, no. 2 (2011): 173.

[15] K. Ron. "Scaling up the accuracy of naive-Bayes classifiers: A decision-tree hybrid." In *KDD*, pp. 202-207. 1996.

[16] Microsoft Press, "Microsoft® Computer Dictionary Fifth Edition", Microsoft Press a division of Microsoft corporation, one Microsoft way, Redmond, Washington 980852-6399, ISBN: 0-7356-1495-4, May 01, 2002. (Available at http://www.assaabloyamericasuniversity.com/Other/AssaAbloyAmericasUniv/Library/Reference%20Docs/Computer%20Dictionary.pdf).

[17] I. Anish, Q. Mohamand, N. Gille, H. Mohamad, "P2P simulator for Queries Routing Using Data Mining", *International Journal of Database Management System (IJDMS)*, Vol.3, No.3, DOI: 10.5121/ijdms.2011.3301, August 2011.

[18] http://www.cs.waikato.ac.nz/ml/weka/downloading.html (Visited on May 24, 2013).

[19] Ian H. Witten, Eibe Frank, Mark A Hall, "Data Mining: Practical Machine Learning Tools and Techniques Third Edition", Morgan Kaufmann Publishers, an imprint of Elsevier, 30, Corporate Drive, Suite 400, Burlington, MA 01803, USA, ISBN: 978-0-12-374856-0, 2011.

[20] http://www.cs.waikato.ac.nz/ml/weka/arff.html (Visited on May 24, 2013).

[21] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications", in: *Proceedings of the 2001 Conference on applications, technologies,*

architectures and protocols for computer communications, S. 149–160, ACM Press, 2001.

[22] S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Schenker, "A scalable content-addressable network", in: *Proceedings of ACM SIGCOMM 2001*, S. 161– 172. ACM Press, 2001.

[23] A. Rowstron, P. Druschel, "Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems", in: *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, S. 329–350, 2001.

[24] E. Buchmann, K. Bohm, "How to Run Experiments with Large Peer-to-Peer Data Structures", in: *Proceedings of the 18th International Parallel and Distributed Processing Symposium*, Santa Fe, USA, April 2004.

[25] K. Aberer, P. Cudre-Mauroux, M. Hauswirth, T.V. Pelt, "Gridvine: Building internet-scale semantic overlay networks", Technical report, EPFL, 2004.

[26] S. Chakrabarti, "Mining the Web: Discovering Knowledge from Hypertext Data" Morgan Kaufmann, San Francisco, 2002.

[27] Y. Wang, L. Galanis, D.J. de Witt, "Galanx: An efficient peer-to-peer search engine system", Available at http://www.cs.wisc.edu/yuanwang, 2003.

[28] S. Datta, K. Bhaduri, C. Giannella, R. Wolff, H. Kargupta, "Distributed data mining in peer-to-peer networks", *Internet Computing, IEEE*, 2006.

[29] F. Dewan Md and R. Mohammad Zahidur, "Attribute weighting with adaptive NBTree for reducing false positives in intrusion detection." *arXiv preprint arXiv:1005.0919* (2010).

[30] E. Mete, and E. Saban, "Implementation and comparison of machine learning classifiers for information security risk analysis of a human resources department." In *Computer Information Systems and Industrial Management Applications (CISIM), 2010 International Conference on*, pp. 187-192. IEEE, 2010.

[31] H. Ahmed et al. "The importance of handling multivariate attributes in the identification of heart valve diseases using heart signals." In *Computer Science and Information Systems (FedCSIS), 2012 Federated Conference on*, pp. 75-

79. IEEE, 2012.

[32] S. Bekti Maryuni. "Naïve Bayes Decision Tree Hybrid Approach for Intrusion Detection System." *Bulletin of Electrical Engineering and Informatics*2, no. 3 (2013).