A New Model to Enhance Efficiency in Distributed Data Mining Using Mobile Agent

Saeed Ngmaldin Bardab[1], Tarig Mohamed Ahmed[2,3]
Tmahmad@Kau.edu.sa
[1]Department of Computer Sciences,ALNeelain University, Sudan-mail: Sa_ngm09@yahoo.com
[2]Assoc. Prof -Department of IT, King Abdul-Aziz University, KSA,
[3]Department of computer Sciences, University of Khartoum, Sudan

Abstract

As a result of the vast amount of data that is geographically found in different locations. Distributed data mining (DDM) has taken a center stage in data mining. The use of mobile agents to enhance efficiency in DDM has gained the attention of industries, commerce and academia because it offers serious suggestions on how to solve inherent problems associated with DDM. In this paper, a novel DDM model has been proposed by using a mobile agent to enhance efficiency. The main idea behind the model is to use the Naive Bayes algorithm to give the mobile agent the ability to learn, compare, get and store the results on it from each server which has different datasets and we found that the accuracy increased roughly by 0.9% which is our main target.

Keywords:
Distributed data mining, Mobile Agent System.

.1. Introduction

The ever-expanding need for relevant data in decision making has created immense interest in data mining, which is the extraction of necessary information from a collection of data using dataset methods such as classification, association rule learning regression, clustering, time series analysis, summarization, and prediction. Han Pei and Kamber defined data mining as the procedures involved in identifying remarkable patterns and knowledge from huge quantities of data [1]. They list the sources of such data to include the Web, databases, data warehouses and other data depositories. Dunham, however, described data mining as the utilization of algorithms to get information and patterns while following knowledge discovery in databases (KDD) process [2]. According to Roiger data mining is the method of unearthing exciting patterns in data with the aim of employing these patterns to clarify present conduct or to forecast forthcoming outcomes [3]. Algarni stated that data mining involves the discovery, grouping and summarization of the links found in the data to make or enhance decisions [4]. Aggarwal provided a broad definition of data mining consisting of gathering, cleaning, treating, examining, and acquiring valuable knowledge from data [5]. Tan, Steinbach & Kumar warned that not all information discovery can be classified as data mining and they differentiate between data mining and information retrieval. [6]

There are many methods in data mining. Jothi & Husain listed some of these methods: Gaussian mixture; standard support vector data description; k-nearest neighbor; density induced support vector data description; artificial neural network; vector quantization; statistical; discriminant analysis; decision tree; Markov based; swarm intelligence; genetic classifiers; support vector and association rule. [7]

Today, data mining is far more complex than it was years ago. We now have data in sources such as the internet, intranet, local area network (LAN) and other sources which are geographically separated. DDM is about how these decentralized and local sources are centralized into a global system.

In this paper we discussed the background of the distribute data mining in §2, related work in §3, MATERIAL AND METHOD in § 4, Discussion in §5, Conclusion in §6, and future work in §7.

2. Background

2.1 Distributed Data Mining

DDM is a division of data mining that focuses on data from different geographical locations. According to Alotaibi, Abdullah & Mosli, the complete procedure for mining important knowledge from data is knowledge discovery and that data mining is only one stage in the procedure [8]. Zaki observed that DDM covers coupled systems which are loose. He adds that DDM can range from a cluster of workstations over an Ethernet local network to geographically distributed locations over a wide area network [9]. Fu, Y. explained that in DDM, the data sets are which are physically dispersed and stored in local databases are accommodated by local computers connected by a computer network. [10] He added that data mining activities take place at both local and global levels with the local level results being put together to arrive at a global result. Zeng et al. maintained that DDM is a way a preventing the problem of transmitting distributed data to one data center since it is costly, unfeasible and has

Manuscript received March 5, 2021
Manuscript revised March 20, 2021
https://doi.org/10.22937/IJCSNS.2021.21.3.36

bandwidth limitations as well as privacy worries. [11] Devi, on the other hand, felt that processing data in a central data warehouse has costs that include storage, communication and computational costs. He also included the problem of confidential data [12]. Kulkarni et al. agreed that processing data has become more difficult because stored online data doubles annually; datasets are dispersed geographically and the immobile nature of datasets.
The use of DDM in data mining has become common in all sectors of social, economic and political activities [13]. Accordingly, Gan et al. concluded that DDM is an influential instrument for the end user, business or political regimes to examine data and learn different kinds of valuable knowledge [14]. Ali et al. remarked the reason why DDM is used is to solve the problems of storage, time and scalability [15]. Chikhale asserted that the goal of DDM is to gather beneficial data, pattern and knowledge so that they can be used in decision making [16].
DDM is seen as a potentially interesting field. Urmela and Nandhini therefore envisaged that DDM will be the most exciting area of data mining in the future because it deals with data situated at diverse geographical sites [17].

2.2 Data Mining Algorithm

Naive Bayes:

Statistically, Naive Bayes classifiers are a group of simple "probabilistic classifiers" founded on Bayes' theorem with strong (naïve) independence suppositions between the features. They are among the simplest Bayesian network models [38].
Naive Bayes is a straightforward method for creating classifiers: models that allot class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some predetermined set. Every naive Bayes classifier presume that the value of a particular characteristic is independent of the value of any other feature, given the class variable. For instance, a vegetable may be considered to be a tomato if it is red, round, and about 3 cm in diameter. To a naive Bayes classifier, each of these features will contribute autonomously to the probability that this vegetable is a tomato, regardless of any possible correlations between the color, roundness, and diameter features. [39]
For some types of probability models, naive Bayes classifiers can be trained very adequately in a supervised learning environment. In many real-world applications, parameter estimation for naive Bayes models uses the method of maximum likelihood; in other words, one can perform tasks with the naive Bayes model while rejecting Bayesian probability or using any Bayesian methods at the same time.
The original Naïve Bayesian technique is based on the conditional probability and the maximum likelihood occurrence. The Naive Bayesian algorithm is given as follows [40]:
Let G be the training set with N tuples where each tuple is represented as a 'k' dimensional attribute vector M, where M={M1 ,M2 ,…Mk }
Let there be 'p' classes C1 , C2 ,…Cp . According to this Naive Bayesian classifier, a tuple T belongs to class Cx only when it has a higher conditional probability than any other class
Cy , where x ≠ y. P(Cx | T) > P(Cy | T) and P(Cx | T) = (P(T | Cx * P(Cx)) / P(T)
Since class conditional independence is assumed,
P(M | Cx) = =1 P k i P(Mk | Cx) = P(M1)* P(M2) * P(M3) …. * P(Mk)
Class Cx is predicted as the output class when
P(M | Cx) * P(Cx) > P(M | Cy) * P(Cy), where 1 ≤ x,y ≥ p and x
≠ y

Algorithm flowchart

The following flowchart explain Naïve Bayes algorithm

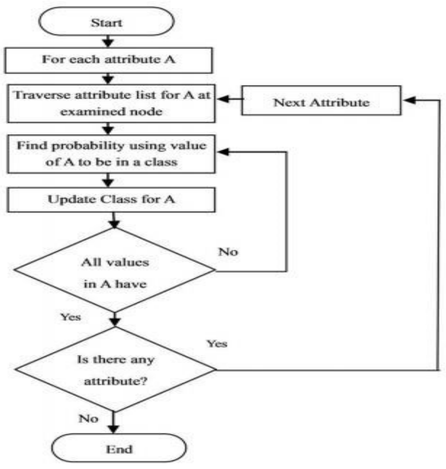


Figure 1 Naive bayes algorithm flowchart

2.3 Mobile Agents

Mobile agents are software programmers that travel autonomously to remote destinations to perform specific tasks for its user. Sahai and Morin referred to mobile agents as self-reliant programmers that roam a network by moving among computers or other mechanisms joined to computer networks, controlling their own actions and collaborating with other computers, data and agents [18].

Mobile Agent System:

Mobile agents are software programmers that travel autonomously to remote destinations to perform specific tasks for its user. Sahai and Morin referred to mobile agents as self-reliant programmers that roam a network by moving among computers or other mechanisms joined to computer networks, controlling their own actions and collaborating with other computers, data and agents [18]. Das explained that the central task for mobile agents involve traversing in a graph by taking a minimum of one trip to every node in an orderly way. Mobile agents have been a worthy replacement of the client/server paradigm because of the limitations of the latter [33]. In addition, mobile agents possess many advantages that make them ideal for distributed systems. Satoh wrote that these advantages include the reduction of communication costs; their ability to execute tasks asynchronously; and their ability to decide their destinations and dynamically deploy their code and data [34]. Das admitted that the use of mobile agents is prevalent on the internet and adds that examples of their usage include tasks gathering data, discussing business transactions and online shopping [35]. According to Lange, the immense benefits of mobile agents in distributed systems are what should be of interest to us. He therefore lists the following benefits of mobile agents: lessening of network load; solving the problem of network latency; protocols encapsulation; dynamically and autonomously adjusting to changes; heterogenous; and resistance to faults because of their strength [36]. It is a strength that is derived from their unique features which Griss and Pour have listed as follows: autonomous, knowledgeable, mobile, adaptable, persistent and collaborative [37].

3.Related Work

A large number of existing studies in the broader literature have examined multi agents being used in distributed data mining to enhance efficiency. (see Moghadam & Ravanmehr, [19]; Aswanandini, [20]; Nure, [21]; Devasekhar & Natarajan, [22]; Bosse, [23]; Hemamalini & Mary, [24]). However, there is also considerable work on the deployment of mobile agents in distributed data mining. Wu et al. offered a genetic algorithm to calculate an imprecise answer to the problem quality and cost of fused data by appropriately introducing a two-tier encoding system and genetic operatives modified to the proposed role. Their result after a simulation shows better algorithm performance over two current heuristics, which are local and global closest first approaches. [25]
Khan and Mohamudally discussed the application of traditional algorithms in data mining with mobile agents in distributed systems. They deliberate on the most widely used data mining algorithm and 'Diabetes' and 'Breast Cancer' were chosen as medical datasets. Their results were satisfactory when data mining goals are considered. [26]
Bosse and Engel proposed combining crowdsensing and social data mining in simulation worlds by using mobile agents to achieve augmented virtually for ultramodern smart cities. Bosse & Engel stated that mobile agents are used here because of their ability to perform certain functions [27]. They list some of these functions as: performing crowdsensing by playing a role in answering questions through a chat blog using a WEB App that is made available to many people. Furthermore, mobile agents can serve as sensor aggregators used to execute sensor fusion in a wide area. The framework of this proposal consists of the key element of the distributed agent processing system which is the JavaScript Agent Machine (JAM) and it is made up of one or more simulators. The Simulation Environment for JAM (SEJAM) gives virtualization with extra simulation regulation and visualization levels up. An agent in a JAM system is allotted to a particular virtual JAM (vJAM) instance with every vJAM having its own state. Every JAM node is recognized by an arbitrarily made node name using phonemes generators. Bosse & Engel concluded that mobile agents are deployed to blend physical and simulated worlds thereby realizing augmented virtuality. They insist that using JAM agent processing platform permits massive recreation of lengthy and intricate socio-technical systems like crowdsensing .
Joshi et al. proposed a frequent pattern mining for distributed databases using mobile agents. They suggest MADFPM Algorithm with a system that transfers item sets only once and sidesteps the numerous transfers of item sets. This method possesses a chief element termed Parent that works at a main location. It sends mobile agents it has made to distant distributed sites, collect outcomes obtained from other mobile agents, and work on them. The Parent sends these agents two times. In the beginning, it sends the mobile agents known as Support Count Mobile Agent to dispersed places to collect the Local Support Count (LSC) of the items. Accordingly, it calculates Global Support Count (GSC) by totaling the LSCs of corresponding items taken from distributed locations. After that, an L-order is calculated from the computed GSCs of the items. In its second mission, the Parent sends the mobile agents, FP-tree Mobile Agents to all the distributed sites with an L-order and collect frequent item sets. The mobile agents also create FP-tree by using the distributed database found there.
The mobile agents then return to the Parent with Local Disjoint Matrix (LDM) which has "Composite Item sets".
Joshi et al. concluded that the algorithm MADFPM for frequent pattern mining of distributed databases using mobile agents is better than the client-server paradigm because the pre-processed compact data is sent to the central site rather than moving the whole data to the central location and then treating it there. They also conclude that MADFPM is better than other mobile agent-based systems like PMFI and PMFI-A, because it cuts down on the computational cost and network traffic [28].
Dukitha and Banumathi compared intelligent agent techniques for distributed data databases as well as a location selection for DDM using density estimation. Their work is under the premise that intelligent agent concepts

have been used in decisive support systems (DSS) to find solutions to enormous conventional problems through the application of DDM. Dukitha and Banumathi's reference of Multi Agent Systems include mobile agents and is made up of other agents such as users interface agent, knowledge management agent, task management agent, communication agent, data mining agent, rule definer agent, knowledge discovery agent and preprocessing agent. They explain that these systems are also categorized into four modules: user module, management module, a processing module and resource module. In their work, the decision making process of the DDM model has these stages: agent initialization; DDF-Distributed Data Fetching; pre-processing; and classification of data for decision making. In addition, Dukitha and Banumathi also proposed an intelligent agent to select the group of locations where the data can be recovered. The result shows that when intelligent agents are used for distributed data mining, decisions are more accurate and less time is needed [29].

Urmela and Nandhini presented comprehensive study of the methods and procedures that work with DDM. They mention that DDM has two architectural models namely client-server based DDM and agent-based DDM architecture. According to Urmela and Nandhini, the agent based DDM can also be divided into two: stationary-based agent DDM architecture and mobile-based agent DDM architecture. They concluded that the agent based DDM has the advantage of reduced space complexity and computation time because data is collected and treated at the local level [17].

Raja and Raja discussed a new novel framework which enhances the efficiency of mobile agents for distributed association rule mining (MADARM) with parallel access of data from distributed sites' database through the application of mobile agents. They suggested a Mobile Agent based Distributed and Parallel Association Rule Mining (MADPARM) on transactional data which is constructed on IBM's Aglets Workbench System. This MADPARM is also created using MADARM with parallel access of distributed sites by deploying mobile agents to improve the agent's interaction with Knowledge Server (KS) which also boosts the FI mining process by applying CBT-fi algorithm in the distributed sites. Their experimental results indicate that MADPARM functions better than the CBT-fi algorithm based MADARM [30].

Ali and Bagchi discussed a design of software architecture for mobile agent-based load monitoring system with a design founded on a probabilistic normed estimation model and related monitoring algorithms. Their projected architecture possesses three key modules: (a) Agent Monitoring Module (AMM), (b) Agent Decision Module (ADM) and, (c) Agent Migration Module (AMMO). The agent monitoring module is used to gather the existing standing of the current resources in the distributed systems. The internals of the architecture of their planned monitoring system is rely on mobile agents. They also presented a comprehensive qualitative and quantitative scrutiny of several mobile agent models. Their experimental evaluations and scalability analysis demonstrate the nature of agents and the capability of the system under varied load situations. They conclude that their projected method lessens the waiting time of a node and the network load which therefore improves the overall system performance [31].

Urra and Ilarri suggested a technique that uses mobile agents to examine data in vehicular systems by means of spatial crowdsourcing. Their model involves the use of mobile agents to scour and screen data that are taken by different vehicles on the streets with their sensors. Their results demonstrate that spatial crowdsourcing technique can improve the monitoring process by permitting the mobile agent to arrive at the needed location faster. Furthermore, their proposed model offers a more effective use of the wireless communications bandwidth and has a faster data collection rate than other models which do not use spatial crowdsourcing [32].

4.MATERIAL AND METHOD

4.1 Model Framework

Agent architecture consist of ***Agent, Sender and*** 4 ***Servers.*** The scenario views the mobile agent system through a simulation that uses multi-threading, each thread (execution path of a program) represents a ***Server*** each host run in two modes, first as a sender in which the host creates an ***Object*** instance of the ***Agent*** and serialize it in order to be sent to the next ***Server***, the next ***Server*** (as a Receiver) listens to a TCP request and DE serialize (translates) it into ***Bytes array*** and sends it to the next ***Server***. The route for the agent is set at the first ***Server*** called ***Sender Server*** and the current location of the ***Agent*** is kept at the ***Agent Object as follows in*** Figure 4 Mobile agent graph**.**

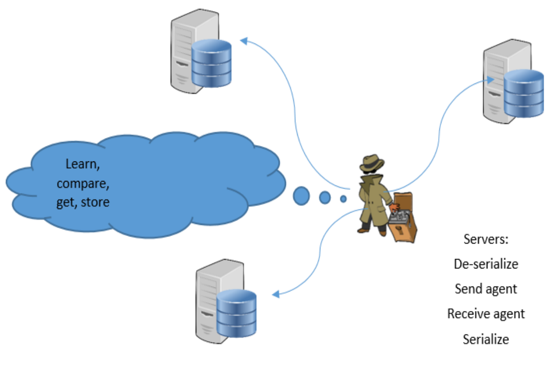


Figure 2 Mobile agent graph

The Mobile Agent System is simulated using C#. The project contains 6 classes; Sender server class, Agent class and 4 Virtual Servers classes.

Agent class:
It works as the management class it holds the current location of our Agent an Agent class contains:

a) agent name: a variable that keeps hold of Agent name.
b) route []: an array to keep hold of all ports assigned by the first initiation.
c) location: a variable that keeps hold of the current route [] index.

Server Sender class:
It initiates the first *agent instance* and stores all **Servers** ports in an array *route []* and then serialize the agent object to be transmitted through network stream.

Server Sender class contains:

Server class:
A *Server* receives the transmitted data through *Socket* and desterilizes it using binary formatter into an array of bytes. Then after receiving complete the data is serialized and streamed through network onto next *Server.*

A *Server* class runs two **functions**:
1) c*reate_agent()* function : which creates a store for the data held by *Agents* and desterilize it into a string of bytes.
2) send_agent(agent instance, location):

which accepts two parameters *agent_instance* = our agent instance, *location* = the next location on *route*. The function creates a network stream by connecting to a *localhost* and *port* number through TCP Client then serializes the agent data into array of bytes, then transmitting data through network stream.

The **function** contains following methods

a) TCP Client: provides simple methods for connecting, sending, and receiving stream data over a network in synchronous blocking mode.
b) Network Stream: Provides the underlying stream of data for network access.
c) Binary Formatter: Serializes and desterilizes an object, or an entire graph of connected objects, in binary format.

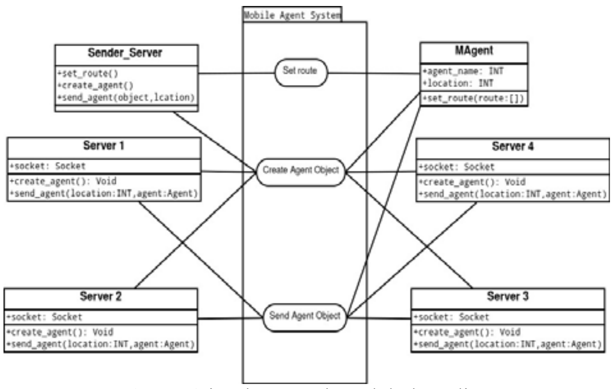


Figure 3 implemented model class diagram

4.2 Data Collection

The datasets for diabetes patients in USA, included 50 criteria over 10 years from (1999-2008) demonstrate the medical records.

4.2 Data Preprocessing

28 attributes were selected (28 attributes and 10061 instances) to generate the proposed model after completion of the preprocessing task.
Data is distributed between 3 hosts with different splitting ratios as follows:

Host-1
Dataset on host 1 containing 50% as training set and 50% as testing set Error! Reference source not found

Table 1 Dataset on host 1

Training Set Size	Test Set Size
5031	5030

Host-2
Dataset on host 2 containing 66% as training set and 33% as testing set Error! Reference source not found

Table 2 Dataset on host 2

Training Set Size	Test Set Size
6707	3354

Host-3
Dataset on host 3 containing 80% as training set and 20% as testing set Error! Reference source not found

Table 3 Dataset on host 3

Training Set Size	Test Set Size
8049	2012

4.4 Tools and Techniques

Python

A widely used programming language in machine learning.

Pandas

Python data framing library used to import, export and manipulate files.

Weka

Analytic framework used for verification purposes.

4.5 Model Implementation

Naive Bayes data mining techniques were selected to develop the model; The Naive Bayes algorithm is based on the conditional independence model of each predictor given the target class. The Bayesian principle is to assign a case to the class that has the largest posterior probability

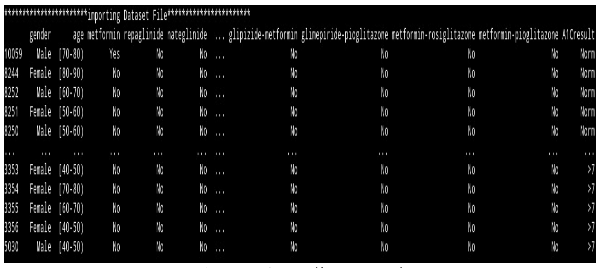
4.5.1 Algorithm Procedure

Naïve Bayesian Algorithm for Diabetes dataset:

```
• Begin
• Initialization nc → Number of classes na → Number of
  attributes N → Number of samples
• for each class Ci do Calculate prior probability
      $P(C_i) = \sum C_i / \sum N, i \in \{1,nc\}$
• for each class Ci do for each attribute Aj do Calculate
  the conditional probability
  $P(A_j|C_i) = \sum C_i \text{ with } A_j / \sum C_i , i \in \{1,..,nc\}$ and $j \in \{1,..,na\}$
• for each class Ci do Calculate the conditional probability
  of the tuple K
  i.e $P(K|C_i) = P(A1|C_i) * P(A2|C_i) * \ldots * P(Ana|C_i)$
• for each class Ci do Calculate the posterior probability
  of the tuple
      K i.e $P(C_i) * P(K|C_i)$
• Prediction If $((P(C_p) * P(K|C_p)) > (P(C_q) * P(K|C_q)))$
  Prediction → C p Else Prediction → Cq Where $p,q \in \{1,..,nc\}$ and $p \neq q$
• End
```

1) **Parsing Dataset:**

Algorithm starts by reading dataset file using pandas as in Figure 4

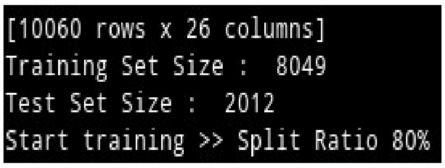


```
**********************importing Dataset File**********************
      gender      age metformin repaglinide nateglinide  ... glipizide-metformin glimepiride-pioglitazone metformin-rosiglitazone metformin-pioglitazone A1Cresult
10059   Male  [70-80)       Yes          No          No  ...                  No                       No                      No                     No      Norm
8244  Female  [80-90)        No          No          No  ...                  No                       No                      No                     No      Norm
8252    Male  [60-70)        No          No          No  ...                  No                       No                      No                     No      Norm
8251  Female  [50-60)        No          No          No  ...                  No                       No                      No                     No      Norm
8250    Male  [50-60)        No          No          No  ...                  No                       No                      No                     No      Norm
...      ...      ...       ...         ...         ...  ...                 ...                      ...                     ...                    ...       ...
3353  Female  [40-50)        No          No          No  ...                  No                       No                      No                     No        >7
3354  Female  [70-80)        No          No          No  ...                  No                       No                      No                     No        >7
3355  Female  [60-70)        No          No          No  ...                  No                       No                      No                     No        >7
3356  Female  [40-50)        No          No          No  ...                  No                       No                      No                     No        >7
5030    Male  [40-50)        No          No          No  ...                  No                       No                      No                     No        >7
```

Figure 4 reading csv dataset

2) **Data Splitting:**

Data is splitted using predefined ratio into training set and testing set as in Figure 5

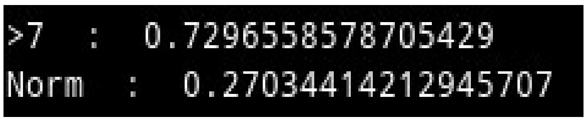


```
[10060 rows x 26 columns]
Training Set Size :  8049
Test Set Size :  2012
Start training >> Split Ratio 80%
```

Figure 5 Data Splitting

3) **Calculations:**

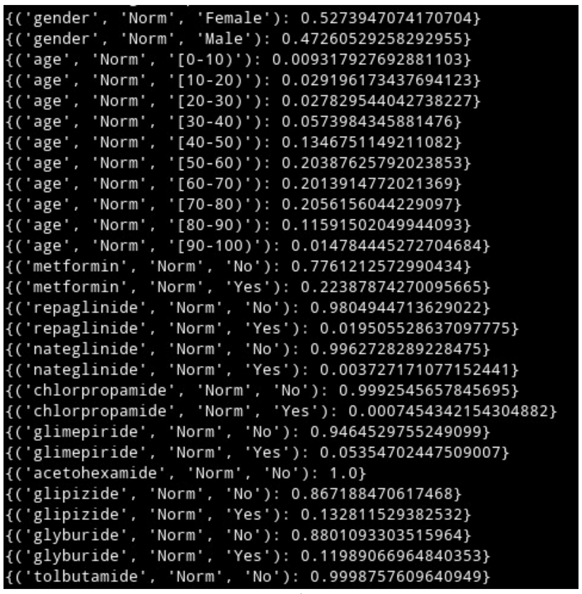
a) prior probability of Class P(C):



```
>7  :  0.7296558578705429
Norm  :  0.27034414212945707
```

Figure 6 prior probability

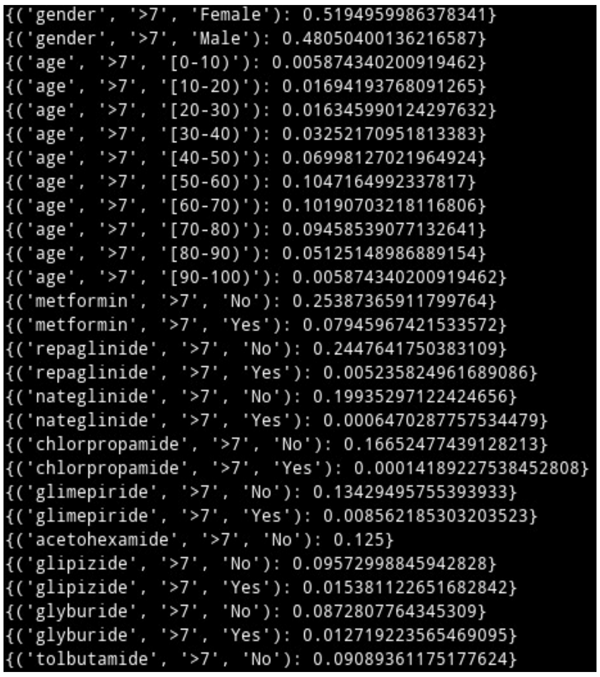
b) prior probability of predictor P(X) :



```
{('gender', 'Norm', 'Female'): 0.5273947074170704}
{('gender', 'Norm', 'Male'): 0.47260529258292955}
{('age', 'Norm', '[0-10)'): 0.009317927692881103}
{('age', 'Norm', '[10-20)'): 0.029196173437694123}
{('age', 'Norm', '[20-30)'): 0.027829544042738227}
{('age', 'Norm', '[30-40)'): 0.0573984345881476}
{('age', 'Norm', '[40-50)'): 0.1346751149211082}
{('age', 'Norm', '[50-60)'): 0.20387625792023853}
{('age', 'Norm', '[60-70)'): 0.2013914772021369}
{('age', 'Norm', '[70-80)'): 0.2056156044229097}
{('age', 'Norm', '[80-90)'): 0.11591502049944093}
{('age', 'Norm', '[90-100)'): 0.014784445272704684}
{('metformin', 'Norm', 'No'): 0.7761212572990434}
{('metformin', 'Norm', 'Yes'): 0.22387874270095665}
{('repaglinide', 'Norm', 'No'): 0.9804944713629022}
{('repaglinide', 'Norm', 'Yes'): 0.019505528637097775}
{('nateglinide', 'Norm', 'No'): 0.9962728289228475}
{('nateglinide', 'Norm', 'Yes'): 0.003727171077152441}
{('chlorpropamide', 'Norm', 'No'): 0.9992545657845695}
{('chlorpropamide', 'Norm', 'Yes'): 0.0007454342154304882}
{('glimepiride', 'Norm', 'No'): 0.9464529755249099}
{('glimepiride', 'Norm', 'Yes'): 0.05354702447509007}
{('acetohexamide', 'Norm', 'No'): 1.0}
{('glipizide', 'Norm', 'No'): 0.867188470617468}
{('glipizide', 'Norm', 'Yes'): 0.132811529382532}
{('glyburide', 'Norm', 'No'): 0.8801093303515964}
{('glyburide', 'Norm', 'Yes'): 0.11989066964840353}
{('tolbutamide', 'Norm', 'No'): 0.9998757609640949}
```

Figure 7

conditional probability “likelihood” P(C|X) :



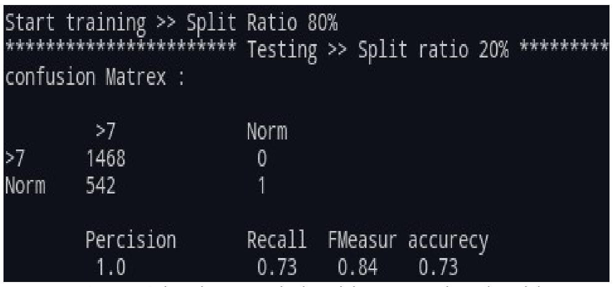
```
{('gender', '>7', 'Female'): 0.5194959986378341}
{('gender', '>7', 'Male'): 0.48050400136216587}
{('age', '>7', '[0-10)'): 0.005874340200919462}
{('age', '>7', '[10-20)'): 0.01694193768091265}
{('age', '>7', '[20-30)'): 0.016345990124297632}
{('age', '>7', '[30-40)'): 0.03252170951813383}
{('age', '>7', '[40-50)'): 0.06998127021964924}
{('age', '>7', '[50-60)'): 0.1047164992337817}
{('age', '>7', '[60-70)'): 0.10190703218116806}
{('age', '>7', '[70-80)'): 0.09458539077132641}
{('age', '>7', '[80-90)'): 0.05125148986889154}
{('age', '>7', '[90-100)'): 0.005874340200919462}
{('metformin', '>7', 'No'): 0.25387365911799764}
{('metformin', '>7', 'Yes'): 0.07945967421533572}
{('repaglinide', '>7', 'No'): 0.2447641750383109}
{('repaglinide', '>7', 'Yes'): 0.005235824961689086}
{('nateglinide', '>7', 'No'): 0.19935297122424656}
{('nateglinide', '>7', 'Yes'): 0.0006470287757534479}
{('chlorpropamide', '>7', 'No'): 0.16652477439128213}
{('chlorpropamide', '>7', 'Yes'): 0.00014189227538452808}
{('glimepiride', '>7', 'No'): 0.13429495755393933}
{('glimepiride', '>7', 'Yes'): 0.008562185303203523}
{('acetohexamide', '>7', 'No'): 0.125}
{('glipizide', '>7', 'No'): 0.09572998845942828}
{('glipizide', '>7', 'Yes'): 0.015381122651682842}
{('glyburide', '>7', 'No'): 0.0872807764345309}
{('glyburide', '>7', 'Yes'): 0.012719223565469095}
{('tolbutamide', '>7', 'No'): 0.09089361175177624}
```

Figure 8

4.5.2 Traditional Model

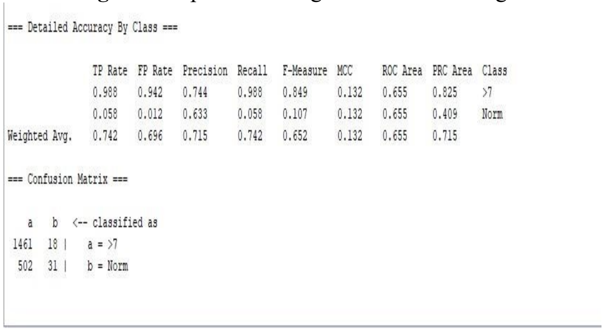
First approach:

Comparing algorithm using entire dataset with weka: Running the above algorithm on entire dataset gave the results



```
Start training >> Split Ratio 80%
********************** Testing >> Split ratio 20% *********
confusion Matrex :

        >7          Norm
>7      1468         0
Norm    542          1

        Percision       Recall  FMeasur accurecy
         1.0             0.73    0.84    0.73
```

Figure 9 implemented algorithm on entire algorithm

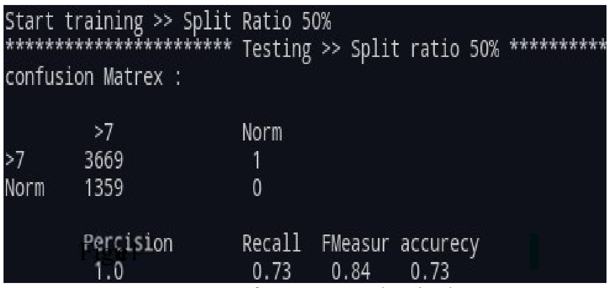


```
=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.988    0.942    0.744      0.988    0.849      0.132    0.655     0.825     >7
                 0.058    0.012    0.633      0.058    0.107      0.132    0.655     0.409     Norm
Weighted Avg.    0.742    0.696    0.715      0.742    0.652      0.132    0.655     0.715

=== Confusion Matrix ===

    a    b   <-- classified as
 1461   18 |    a = >7
  502   31 |    b = Norm
```

Figure 10 weka performance on the entire dataset

Second approach:

Distribution of dataset compared to single dataset:
single server:
An experiment were made using implemented algorithm on a single server using the entire dataset for training and testing with split ratios (66% used as training data and 33% used for testing the trained model) .

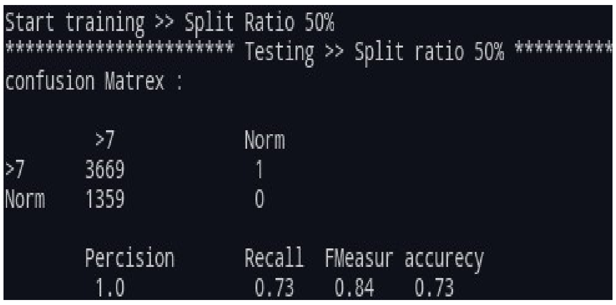


```
Start training >> Split Ratio 50%
********************** Testing >> Split ratio 50% **********
confusion Matrex :

        >7          Norm
>7      3669         1
Norm    1359         0

        Percision       Recall  FMeasur accurecy
         1.0             0.73    0.84    0.73
```

Figure 11 performance on the single server

Third approach:

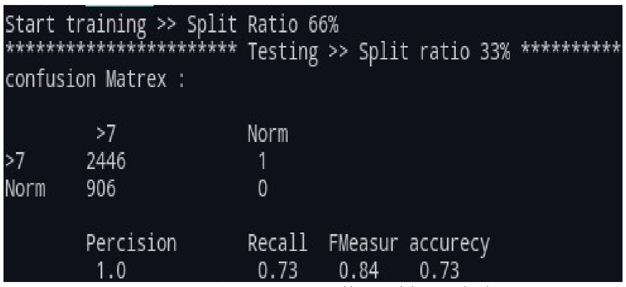
Using mobile agent feature and machine learning on distributed dataset:
By using the model described in Figure 5 implemented model class diagram running the algorithm on Servers mapped as in Fig. 4 the Agent Object travels and visits the first Server viewing collected knowledge.



```
Start training >> Split Ratio 50%
********************** Testing >> Split ratio 50% **********
confusion Matrex :

        >7          Norm
>7      3669         1
Norm    1359         0

        Percision       Recall  FMeasur accurecy
         1.0             0.73    0.84    0.73
```

Figure 12 Server 1 collected knowledge

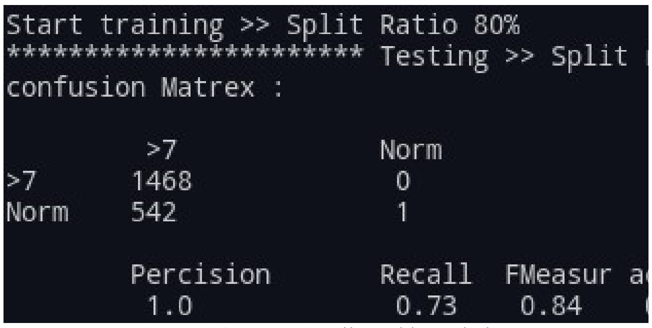
An instance of Agent Object is transmitted through network stream onto the next Server carrying the collected knowledge represented.



```
Start training >> Split Ratio 66%
********************** Testing >> Split ratio 33% **********
confusion Matrex :

        >7          Norm
>7      2446         1
Norm    906          0

        Percision       Recall  FMeasur accurecy
         1.0             0.73    0.84    0.73
```

Figure 13 Server 2 collected knowledge

An instance of Agent Object is transmitted through network stream onto the next Server carrying the collected knowledge represented.



```
Start training >> Split Ratio 80%
*********************** Testing >> Split
confusion Matrex :

        >7             Norm
>7      1468            0
Norm    542             1

        Percision      Recall  FMeasur a
         1.0            0.73    0.84
```

Figure 14 Server 3 collected knowledge

4.5.3 A New Model to Enhance Efficiency in Distributed Data Mining with Mobile Agent

The aim of the proposed new system is the classification and exploration of distributed data taking advantage of the characteristics of the Mobile Agent in mobility and using Machine learning through the use of the Gaussian Naïve Bayes classification algorithm, therefore it will be considered a new model in classification in Distributed Data Mining. The following Model class explain the new Model.

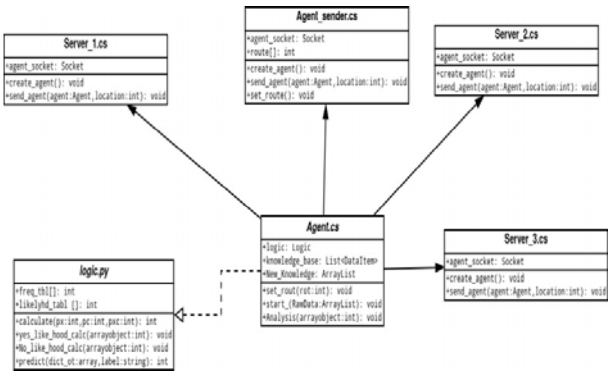

Server_1.cs
+agent_socket: Socket
+create_agent(): void
+send_agent(agent:Agent,location:int): void
Agent_sender.cs
+agent_socket: Socket
+route[]: int
+create_agent(): void
+send_agent(agent:Agent,location:int): void
+set_route(): void
Server_2.cs
+agent_socket: Socket
+create_agent(): void
+send_agent(agent:Agent,location:int): void
Agent.cs
+logic: Logic
+knowledge_base: List<DataItem>
+New_Knowledge: ArrayList
+set_rout(rot:int): void
+start_(RawData:ArrayList): void
+Analysis(arrayobject:int): void
Server_3.cs
+agent_socket: Socket
+create_agent(): void
+send_agent(agent:Agent,location:int): void
logic.py
+freq_tbl[]: int
+likelyhd_tabl []: int
+calculate(px:int,pc:int,pxc:int): int
+yes_like_hood_calc(arrayobject:int): void
+No_like_hood_calc(arrayobject:int): void
+predict(dict_ot:array,label:string): int

Figure 15 Model Class of Proposed Model

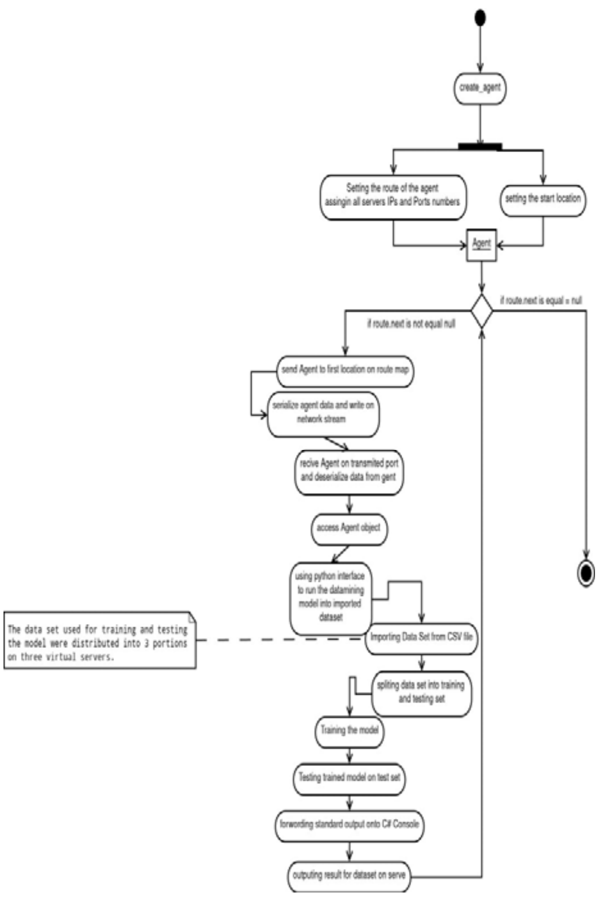
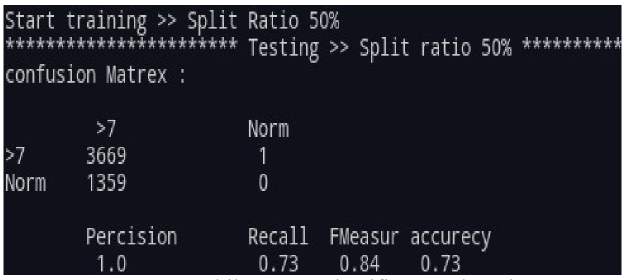

create_agent
Setting the route of the agent assingin all servers IPs and Ports numbers
setting the start location
Agent
if route.next is equal = null
if route.next is not equal null
send Agent to first location on route map
serialize agent data and write on network stream
recive Agent on transmited port and deserialize data from gent
access Agent object
using python interface to run the datamining model into imported dataset
The data set used for training and testing the model were distributed into 3 portions on three virtual servers.
Importing Data Set from CSV file
spliting data set into training and testing set
Training the model
Testing trained model on test set
forwording standard output onto C# Console
outputing result for dataset on serve

Figure *16* Activity diagram of proposed model

Experiment on Host 1

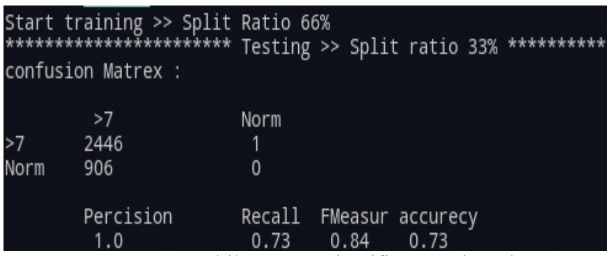
The Host 1 experiment were made on 50% training set



```
Start training >> Split Ratio 50%
*********************** Testing >> Split ratio 50% **********
confusion Matrex :

        >7             Norm
>7      3669            1
Norm    1359            0

        Percision      Recall  FMeasur accurecy
         1.0            0.73    0.84    0.73
```

Figure 17 Mobile Agent Classifier Result on host 1

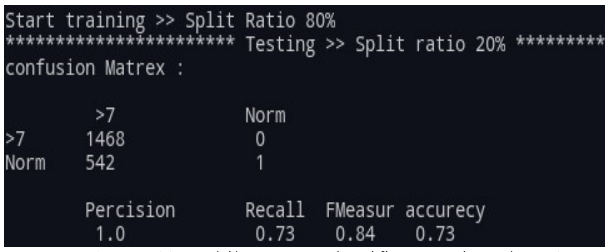
Experiment on Host 2
The Host 2 experiment were made on 66% training set



```
Start training >> Split Ratio 66%
*********************** Testing >> Split ratio 33% **********
confusion Matrex :

        >7            Norm
>7     2446            1
Norm    906            0

        Percision      Recall  FMeasur accurecy
         1.0            0.73    0.84    0.73
```

Figure 18 Mobile Agent Classifier Result on host 2

Experiment on Host 3
The Host 3 experiment were made on 80% training set



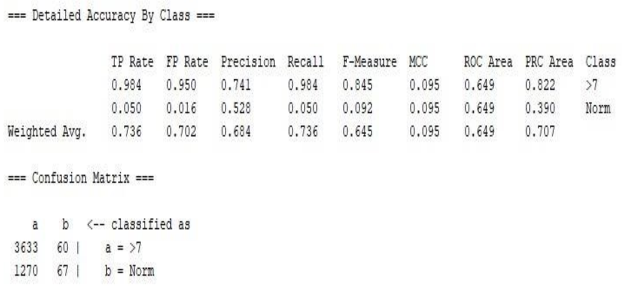
```
Start training >> Split Ratio 80%
*********************** Testing >> Split ratio 20% *********
confusion Matrex :

        >7            Norm
>7     1468            0
Norm    542            1

        Percision      Recall  FMeasur accurecy
         1.0            0.73    0.84    0.73
```

Figure 19 Mobile Agent Classifier Result on host 3

Viewing above results shows that the new model presents a better accuracy when combined with mobile agent feature.

4.5.3 Weka Model

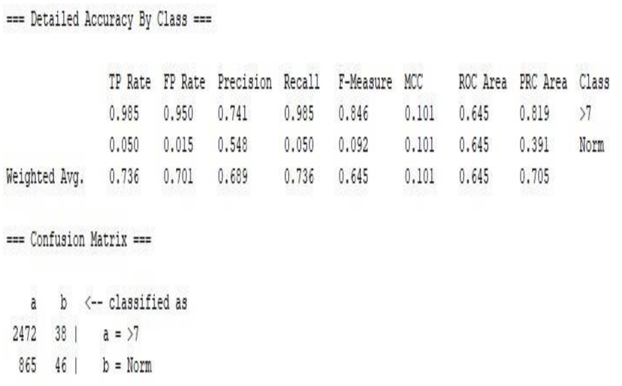
Experiment on Host 1
The Host 1 experiment were made on 50% training set



```
=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.984    0.950    0.741      0.984    0.845      0.095    0.649     0.822     >7
                 0.050    0.016    0.528      0.050    0.092      0.095    0.649     0.390     Norm
Weighted Avg.    0.736    0.702    0.684      0.736    0.645      0.095    0.649     0.707

=== Confusion Matrix ===

    a    b   <-- classified as
 3633   60 |    a = >7
 1270   67 |    b = Norm
```

Figure 20 Weka Classifier Results on host 1

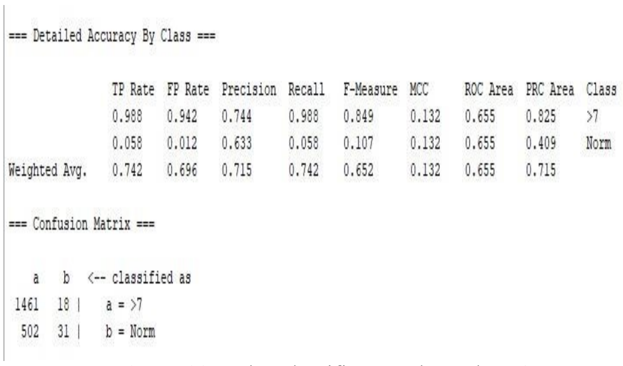
Experiment on Host 2
The Host 2 experiment were made on 66% training set



```
=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.985    0.950    0.741      0.985    0.846      0.101    0.645     0.819     >7
                 0.050    0.015    0.548      0.050    0.092      0.101    0.645     0.391     Norm
Weighted Avg.    0.736    0.701    0.689      0.736    0.645      0.101    0.645     0.705

=== Confusion Matrix ===

    a    b   <-- classified as
 2472   38 |    a = >7
  865   46 |    b = Norm
```

Figure 21 Weka Classifier Results on host 2

Experiment on Host 3
The Host 3 experiment were made on 80% training set



```
=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.988    0.942    0.744      0.988    0.849      0.132    0.655     0.825     >7
                 0.058    0.012    0.633      0.058    0.107      0.132    0.655     0.409     Norm
Weighted Avg.    0.742    0.696    0.715      0.742    0.652      0.132    0.655     0.715

=== Confusion Matrix ===

    a    b   <-- classified as
 1461   18 |    a = >7
  502   31 |    b = Norm
```

Figure 22 Weka Classifier Results on host 3

5. Discussion

According to the above results, the proposed model has better servers' results than the WEKA results. To arrive at a precision measure, we use the formula: Precision = TP/(FN+TP). The precision measure for our model is 1.0 which is better than the WEKA precision measure. We also calculated the recall measure using the formula: Recall = TP/ (FP + TP) and our model recall was 0.73 which is close to the WEKA results. The formula we used for accuracy measure was: Accuracy = TP/ (TP + FN). Accordingly, our accuracy and F- measures were 0.84 and 0.73 respectively which were also better than the WEKA results

Model Results

Table 4 New Model Results

C-Matrix	Server 1	Server 2	Server 3
TP	1468	2446	3669
TN	1	0	0
FP	0	1	1
FN	542	906	1359
Precision	1.0	1.0	1.0
Recall	0.73	0.73	0.73
F-Measure	0.84	0.84	0.84
Accurcy	0.73	0.73	0.73

*TP = True “>7” | TN= True “Norm” | FP = False “>7” | FN= False “Norm”.

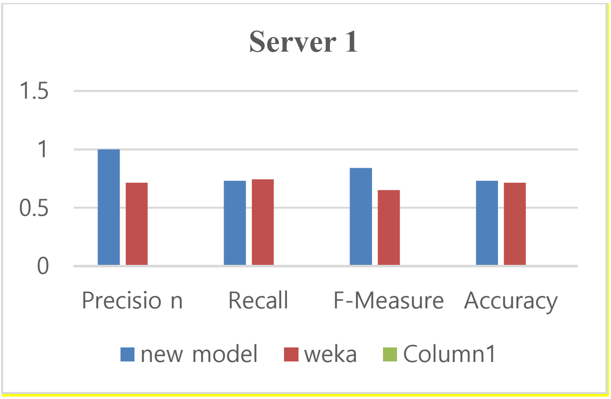
WEKA Results

Table 5 Weka Model Results

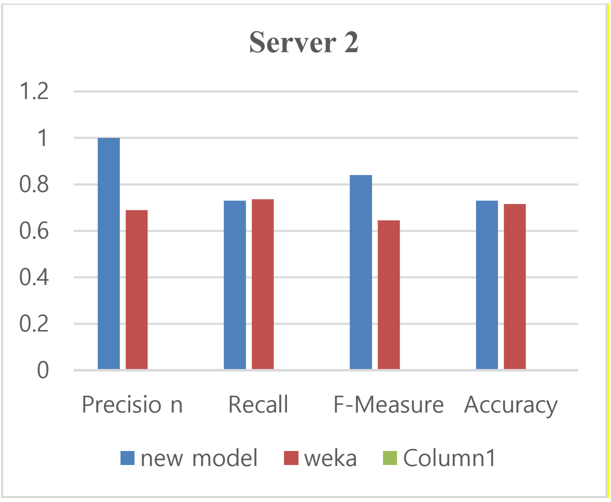
C-Matrix	Split Ratio		
	80%	33%	50%
TP	1461	2476	3633
TN	31	64	67
FP	18	38	60
FN	502	865	1270
Precision	0.715	0.689	0.684
Recall	0.742	0.736	0.736
F-Measure	0.652	0.645	0.645
Accurcy	0.715	0.705	0.717

6. Conclusion

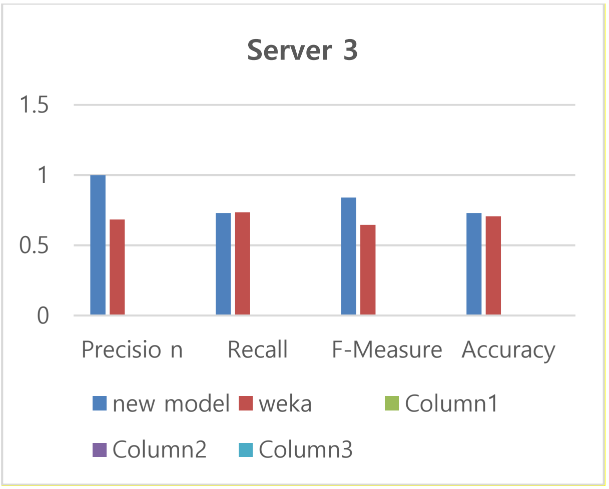
Finally, these studies span various topics such as Distributed Data Mining, Mobile Agent System, Naive Bayes Algorithm, in this dissertation, we focused on one of such topic namely A New Model to Enhance Efficiency in DDM Using Mobile Agent.
We developed A New Model to Enhance Efficiency in DDM by let the MA take the decision and memorized the best results from its own classification algorithm that programed on it. However, our studies focused on the Feature of Mobile Agent with Distributed Data Mining by using Naive Bayes for providing Adaptive learning / knowledgeable to produce a better model.



Graph 1 Confusion matrix Comparison between the new model and weka in server1

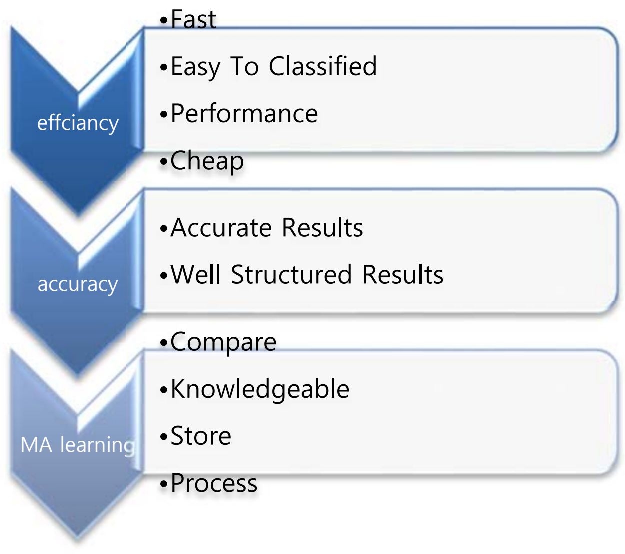


Graph 2 Confusion matrix Comparison between the new model and weka in server2



Graph 3 Confusion matrix Comparison between the new model and weka in server3

Distributed and other features are used with heterogeneous computers and networks. Data mining has been of immense benefit to businesses and government departments.



Graph 4 New model contribution in DDM

7. Future work

Extra future work highly recommended as this field is important and very sensitive for businesses, which will open grate opportunities in DDM. Also a real implementation recommended to get more results to insure the success of this model. MA Security has many issues, so extra work will be need to insure it gives secure results.
In future we will try to create hybrid algorithm or new algorithm that will be able to provide best classification result for every dataset.
This work will be a base for the other future studies.

Reference

[1] J. P. J. &. K. M. Han, Data mining: concepts and techniques, Elsevier, 2011.
[2] M. H. Dunham, Data mining: Introductory and advanced topics, India: Pearson Education India, 2006.
[3] R. J. Roiger, Data Mining: A Tutorial-Based Primer, Second Edition, 2017.
[4] A. Algarni, "Data mining in education," *International Journal of Advanced Computer and Application,* vol. 7, no. 6, pp. 456-461, 2016.
[5] C. C. Aggarwal, Data mining: the textbook, Springer, 2015.
[6] P. N. S. M. &. K. V. Tan, Introduction to data mining, Pearson Education India, 2016.
[7] N. &. H. W. Jothi, Data mining in healthcare–a review, vol. 72, Procedia computer science, 2015, pp. 306-301.
[8] N. M. A. M. &. M. H. Alotaibi, "Agent-based big Data mining," *International Journal of Advanced Trends in Computer Science and Engineering,* vol. 8, no. 1.1, 2019.
[9] M. J. Zaki, Parallel and distributed data mining: An introduction. In large-scale parallel data mining, berlin, heidelberg: Springer, 2000, pp. 1-23.
[10] Y. Fu, "Distributed data mining: An overview," *Newsletter of the IEEE Technical Committee on Distributed Processing,* vol. 4, no. 3, pp. 5-9, 2001.
[11] L. L. L. D. L. L. K. S. Z. W. M. .. &. L. P. Zeng, "Distributed data mining: a survey," in *Information Technology and Management*, vol. 13, 2012, pp. 403-409.
[12] S. G. Devi, "A survey on distributed data mining and its trends," *International Journal of Research in Engineering & Technology,* vol. 2, no. 3, pp. 107-120, 2014.
[13] U. P. T. K. K. M. S. R. &. Y. A. R. Kulkarni, "Exploring the capabilities of mobile agents in distributed data mining," *International Database Engineering and Applications Symposium,* pp. 277-280, 2006.
[14] W. L. J. C. W. C. H. C. &. z. J. Gan, "Data mining in distributes enviromnet," in *Data Mining and Knowledge Discovery*, vol. 7(6), Springer, 2017.
[15] A. A. V. P. K. S. Ž. P. &. U. A. Ali, "Distributed data mining systems: techniques, approaches and algorithms," in *MATEC Web of Conferences. EDP Sciences*, 2018.
[16] M. R. Chikhale, "Study of Distributed Data Mining Algorithm anf Trends," *ISOR Journal of Computer Engineering,* pp. 2278-0661, 2016.
[17] S. &. N. M. Urmela, "Approaches and Techniques of Distributed Data Mining," *International Journal of Engineering and Technology,* vol. 9(1), p. 69, 2017.
[18] C. Sahai M. J. & Morin, "Mobile agent for enabling mobile user aware application," in *Autonomous agents*, 1998.
[19] A. N. &. R. R. Moghadam, "Multi agent distributed mining approach for classifying meteorology data," *International journal of environmental science and technology,* vol. 15(1), pp. 149-158, 2018.
[20] S. Aswanandini, "Survey of security in Multi-Agent System for Distributed Data Mining," *International Journal of Innovative Computer cience & Engineering,* vol. 4(3), 2017.
[21] A. H. Nure, "Distributed data mining using multi-agent data," *International Research Journal of Engineering and Technology,* 2017.
[22] V. &. N. P. Devasekhar, Multi-Agent Distributed Data Mining Challenges And Research Directions, 2020.
[23] S. Bosse, "Industrial Agents and Distributed Agent-based Learning," in *Multidisciplinary Digital Publishing institute Proceedings*, 2016.
[24] R. &. M. L. J. Hemamalini, "n analysis on multi-agent based distributed data mining system," *International Journal of Scientific and Research Publications,* vol. 4, no. 6, pp. 1-6, 2014.
[25] Q. R. N. S. B. J. I. S. S. V. V. K. Q. H. &. C. K. Wu, "On computing mobile agent routes for data fusion in distributed

sensor networks," *IEEE Transactions on Knowledge and Data Engineering,* vol. 16, no. 6, pp. 740-753, 2004.

[26] D. M. &. M. N. Khan, "The adaptability of conventional data mining algorithms through intelligent mobile agents in modern distributed systems," *International Journal of Computer Science Issues,* vol. 9, no. 1, p. 38, 2012.

[27] S. &. E. U. Bosse, "Augmented virtual reality," in *In* Multidisciplinary Digital Publishing Institute Proceedings, 2018.

[28] Y. t. S. G. G. R. B. &. r. P. P. Joshi, "Mobile Agent-Based Frequent Pattern Mining for Distributed Database," in *Intelligent Computing and Information and Communication*, Singapore, Springer, 2018, pp. 77-85.

[29] M. &. B. A. Dukitha, A Comparative Study of Intelligent Agent Techniques for Distributed Data Databases, EasyChair, 2019.

[30] A. &. R. E. Raja, "MADPARM: Mobile Agent based Distributed and Parallel Association Rule Mining," *International Journal of Engineering Trends and Technology,* vol. 49(6), 2017.

[31] M. &. B. S. Ali, "Probabilistic normed load monitoring in large scale distributed system using mobile agent," in *Future Generation Computer Systems*, 2019, pp. 148-167.

[32] O. &. I. S. Urra, "Spatial crowdsourcing with mobile agents in vehicular network," in *Vehicular Communications*, vol. 17, 2019, pp. 10-34.

[33] S. Das, "Graph Explorations with Mobile Agents," in *Distributed Computing by Mobile Entities*, Cham, Springer, 2019, pp. 403422.

[34] I. Satoh, Mobile agents. In Handbook of Ambient Intelligence and Smart Environments, Boston, MA: Springer, 2010, pp. 771791.

[35] S. Das, Mobile agents in distributed computing: Network exploration, vol. 1, bulletin of EATCS, 2013, p. 109.

[36] D. B. Lange, "Mobile objects and mobile agents: The future of distributed computing?," in *In European conference on objectoriented programming*, Berlin, Heidelberg, 1998.

[37] M. L. &. P. G. Griss, Accelerating development with agent components, vol. 34, Computer, 2001, pp. 37-43.

[38] A. McCallum, "Bayesian Network Represention," in *Graphical Models*, 2019.

[39] N. &. T. G. Sneha, "Analysis of diabetes mellitus for early prediction using optimal features selection," in *Journal of Big Data*, 2019.

[40] K. N. &. S. N., "A Naive Bayesian Classifier for Educational Qualification," *Indian Journal of Science and Technology,* vol. 8(16), 2015.