Enhancing Reusability and Measuring Performance Merits of Software Component using Data Mining

G. Maheswari, K. Chitra

Abstract— In software industry, it is necessary to reduce time and efforts in software development. Software reusability is an important measure to improve development and quality of software. Enhancing reusability will reduce delivery time of software products, decreases the development labor, software defects and cost of the development process. Software reuse is the best solution factor to acquire the existing knowledge from software warehouse. Measuring the reusability level of the software is essential to achieve the goals of reuse. Data mining is the process of extracting useful patterns and analyzing data sets from large data sets. The reusability of a software component chooses the right measurement and enhances the preventability of an application for reuse. The software metrics are used as quantitative measures to establish and evaluate the components. In this paper, measuring the software reusability using some classification algorithms on a specific software reuse data set is discussed. The system is implemented using R data mining tool and performance of the automation system is developed for reusability prediction like precision, recall, f-measure. The experimental result shows that the model can be effectively used for efficient, accurate, quicker and economic identification of reusable components from the existing software resources. This paper seeks to provide comparative analysis of H-SOM and Naïve Bayes algorithm classifiers of Dengue datasets.

Index Terms—Software Reusability, Data Mining, Classification, Reusability Prediction, Hierarchical-SOM, Naïve Bayes

1. INTRODUCTION

Software Reuse offers a great deal of potentiality in terms of software productivity and software quality. Software Reusability decreases the time and cost of software development processes. It also increases productivity and quality of software industry. Reusability is an essential feature of a software component. It manages the building, wrapping, pattern, setting up, operation and maintenance issues and supposes these are not able to maintain properly than it appear to be reusable.

Figure 1.1 shows the steps of reusability process. The first step is developed for building a reuse plan and studying the given problem. The next phase is classifying the reuse plan. The construction of the solution is reconfigured to develop the reusability in the upcoming phases. The existing components are acquired, instantiated or modified. The software components are included in the newly developed components and as the last step, the whole product is evaluated and validated.

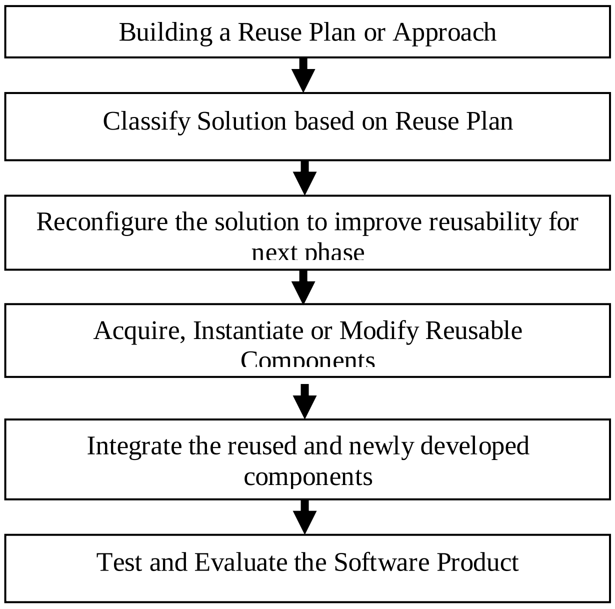

Building a Reuse Plan or Approach
Classify Solution based on Reuse Plan
Reconfigure the solution to improve reusability for next phase
Acquire, Instantiate or Modify Reusable Components
Integrate the reused and newly developed components
Test and Evaluate the Software Product

Figure 1.1 Flowchart of Reusability Process

Nowadays, different data mining techniques are used to extract data from software historical data. Data mining or knowledge discovery is the process of breaking and analyzing enormous sets of data and extracting the meaning of the data. Figure 1.2 shows the knowledge discovery pattern (KDP). The KDP consists of the steps such as selection of data, data cleaning, data transformation, pattern searching (data mining), in addition to discovering appearance, explanation for data and estimation. Data collection phase is used to extract the data related to data mining analysis.

Data cleaning and transformation phase of KDP provides cleaning and preparation of data, converting the data for processing and validating results to get the results. The data mining algorithms are clustering, classification, artificial neural networks, rule association, decision tree and classification and regression trees (CART). The final steps are interpretation and evaluation of results useful for making decisions by applying knowledge discovery techniques.

Revised Version Manuscript Received on April 12, 2019.

G. Maheswari, Assistant Professor, Mangayarkarasi College of Arts and Science for Women, Madurai, Tamil Nadu, India (E-Mail: gm291276@gmail.com)

K. Chitra, Assistant Professor, Government Arts College, Melur, Madurai, Tamil Nadu, India

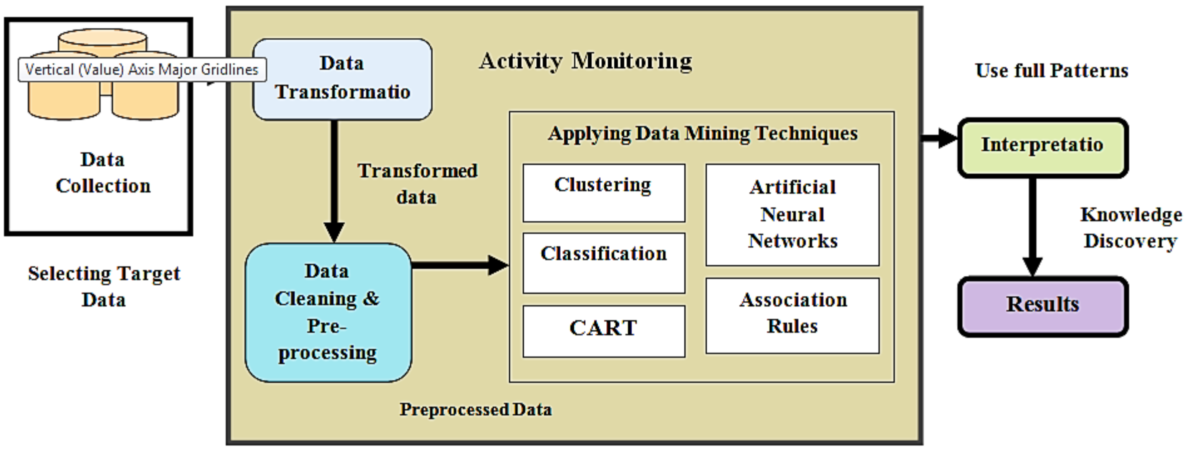


Figure 1.2 Knowledge Discovery Process

NEED OF SOFTWARE REUSE

The software development life cycle (SDLC) is the main phase of software development. The software components build a system instead of designing and developing a new system. Software reuse can be used efficiently to develop the software more quickly with lower budget and within limited period time. Software reuse is mainly divided into process and product reuse. Creation of software components and integrating them into new system is known as Product reuse. Reuse is used for developing efficient software process and repository that produces base knowledge. Thus, reuse process is useful to improve quality, minimizing development reprocess and risk reduction.

BENEFITS OF SOFTWARE REUSE

It **builds** novel software and red**uces** the complexity and dependence of the software products. It is compared with the quality of the software with the existing components. The main advantages of reuse come from the higher level reuse. The reusability attributes are software specifications, development, planning and prototype.

The software reuse has many evident benefits and inevitable problems. The software reuse can reduce costs and time in software developments. The major benefits of software reusability are:

- increase in dependability
- Reduction of Process Risk
- Increase in the Productivity of Software
- Reduction in the operational Cost
- System interoperability enhancement
- Developed with Less Manpower
- Reduces Software Maintenance Cost
- Accelerates Development
- Delivers a good quality software as well as powerful competitive
- Increases effectiveness

As said earlier, the software reuse saves time as well as cost. It resolves the planned development, exploitation and maintenance of reusable software assets. It can be successful because of reduction in the functional product delivery and amount of resource components. Software reusability can improve productivity, time to market, quality of software; it can reduce maintenance cost, allow for system interoperability, risk minimization, knowledge and can increase system functionality.

The rest of this proposed work is illustrated as follows. Section II describes related works about software reusability using data mining. The proposed methodology work and algorithm implementation is described in section III. Section IV presents the experimental results. Section V compromises the conclusion.

2. RELATED WORKS

Tajinder Singh Sondhi et.al(2017) Software reuse is a significant but hidden piece of Software businesses and it provides them with predefined software parts, so as to assemble better software at more rapid and at a lower cost. Reusing software parts help in a quickened, chance free and cost effective advancement of software. However, numerous issues have arrived in more than couple of decades. We have studied various methodologies accessible for software reuse beginning from the traditional ones like REBOOT, faceted execution to the later ones like the PULSE technique, object oriented methodology and plan science research methodologies, pragmatic approach and development model. These systems have continually given better methods that have improved the advancement of time, decreased improvement costs, diminished procedure chance, and improved constancy. Furthermore, they have given an institutionalized way to deal with programming reuse. In this paper, we have compared and analyzed different strategies and techniques accessible for tackling different issues related with the reuse of software parts.

Ahmed Mateen et.al (2017) The high quality software reuse process is useful to improve the reliability, superiority and productivity, and enhance the schedule, cost and evaluation of a software program. The reuse software program and reuse software processes have benefits to decrease risking factors. Software reuse approaches improve the quality in software industry. It increases software quality in less energy and time.

Megha Dabhade et al (2016) argues that several reusability metrics are useful to recognize the software reusability components only. In the reuse level, the software qualities are improved by decreasing the change and rework done at the design phase of the software development that results with better software productivity.

K. Deeba et al (2015) suggested that data mining is the process of analyzing the data and extracting information. Data mining uses several methodologies to predict the relationship between the existing data and the expected outcome results. Classification, clustering, regression, rule generation are the solution to these problems. Classification is a type of data mining algorithm used to create class labels and classify the data, based on sample inputs and classify datasets. The paper focuses on software reusability for classification of data using data mining.

Nilesh Jagdish Vispute (2015) authors discuss data mining as an analysis of huge amount of the data which takes the hidden valuable information from it. Data mining algorithm is used for testing the accuracy in predicting the datasets. The classification consists of predicting certain results based on a given input. To show the outcome of these inputs, the algorithm processes a set of training input with set of attributes and checks with respective outcome, which is called as goal or predictive attribute. The classification of data and outputs were compared using Naïve Bayes, SGD, and H-SOM algorithms and finally simulated the results.

Priyanka kakkar et al (2012) developed most flexible components to calculate and check the reusability of software modules. The purpose of this representation is pattern recognition by finding supervised appearance that helps to evaluate the indescribable aspects of software modules in terms of reusability. There are multiple degrees in reusability of their modules which are used to identify the functionality of software modules. The performance measurements of software components are calculated in terms of volume, coupling, complexity, reuse frequency, regularity and reusability.

Kath et al., (2009) accept that with the move to dispersed, segment based frameworks including reuse of segments and administrations, developing, framework wide properties, incorporating wellbeing and security specifically, are winding up progressively hard to ensure. Model based methods establish a promising way to deal with assurance wellbeing and security in frameworks worked with reusable parts. The key components in this methodology are rightness and certainty by development, and detachment of concerns.

Dantas et al., (2010) discovered with framework improvement ending up progressively steady, programming reuse, and soundness emerges as two of the most alluring properties of top quality software. A key objective in contemporary programming design is advance reuse and stability of the software components concurrently. There are developing number of procedures for improving modularity, running from perspective situated and highlight arranged programming to piece channels. They exhibited an exploratory analysis of advance programming methods on how it is made possible to reach a superior exchange off of programming reuse and solidness. Results revealed that a hybrid manifestation of highlight situated and viewpoint arranged programming is, by all accounts, the most encouraging programming method.

3. THEORETICAL BACKGROUND

PRINCIPLES OF SOFTWARE REUSE

The software reusability principles are characterized in terms of some reuse methodology such as abstraction, selection, integration and specialization. The software reuse is as follows:

1. Develop a software field as structural design and as a support for the reuse behavior.
2. The software development process is used to support and organize reuse.
3. Reuse higher than just code.
4. Practice domain engineering.
5. Incorporate the reuse of a quality management and software engineering actions.
6. Achieve reuse process across product boundaries.
7. Utilize the automation tools to sustain reuse.
8. Concept, content and context.

PROPOSED METHODOLOGY

Software Reusability evaluation for procedure based software components **is done** using following steps:

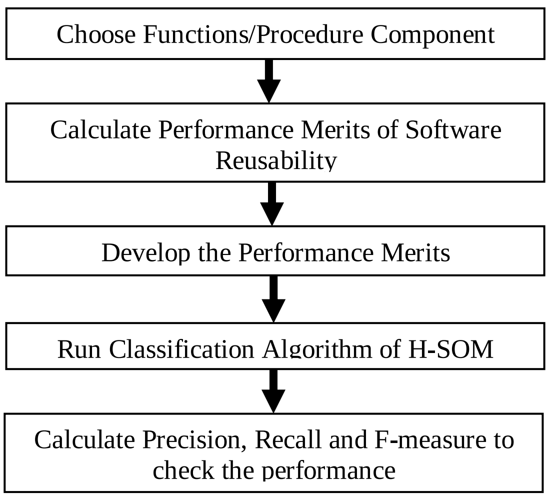

Choose Functions/Procedure Component
Calculate Performance Merits of Software Reusability
Develop the Performance Merits
Run Classification Algorithm of H-SOM
Calculate Precision, Recall and F-measure to check the performance

Figure 1.3 Flowchart of Proposed Methodology

Figure 1.3 shows the flow of the proposed methodology. The proposed model (reusability evaluation tool) data collection is done for some of the classification algorithm (H-SOM).The computed results are evaluated in terms of precision, recall, f-measure. The data mining technique is used to achieve higher software reusability.

NAÏVE BAYESIAN CLASSIFIERS (NB)

NB is a simple Bayesian supervised classifier that assumes that all attributes are independent of each other, given the context of the class:

$$P(x|y=c) = \prod_{i=1}^{D} P(x_i|y) = c \;--------(1)$$

This is called NB assumption which is rarely true in the most real-world situations. NB often performs classification. Due to this presumption, the parameters for each property can be adapted independently, and this greatly simplifies learning, particularly when the number of properties is enormous.

In view of NB presumption, probability of a report given its class can be determined as a result of the probability of the quality values over all word characteristics. Given estimates of parameters can be determined from the training records while classification can be performed on test reports by figuring the back probability of each class given the proof of the test record, and choosing the class with the most elevated probability.

In this model, a record is an arrangement of words taken from a similar vocabulary V and lengths of documents are autonomous of class. Assumption is that the probability of each word occasion in a document is autonomous of the word's setting and position in the document. Thus, each document di is drawn from a multinomial distribution of words and represented in the form of bag of words.

$$P(d_i|C_i;\theta) = P(|d_i|)|d_i|!\prod_{i=1}^{|V|}\frac{P(d_i|C_i;\theta)^{N_{it}}}{N_{it}!} ----(2)$$

The probabilities of each word are parameters of the generative component for each class $\theta_{w_{t|Cj}} = P(w_t|C_i;\theta)$ where $0\leq \theta_{w_{t|Cj}} \leq 1$ and $\sum_t \theta_{w_{t|Cj}} = 1$.

Optimal estimates for these parameters are calculated from a set of labeled training data, and the estimate of the probability of word wt in class cj is:

$$\hat{\theta}_{\,w_{t|Cj}} = P(w_t|C_i;\hat{\theta})\frac{1+\sum_{i=1}^{|D|}N_{it}P(C_j\,|d_i)}{|V|+\sum_{S=1}^{|V|}\sum_{i=1}^{|D|}N_{is}P(C_j\,|d_i)} - (3)$$

Given estimates of parameters are calculated from the training documents; classification can be performed on test documents by calculating the posterior probability of each class given as the evidence of the test document, and selecting the class with the highest probability using Bayes' rule

$$P(C_j|d_i;\hat{\theta})\frac{(C_j|\hat{\theta})P(d_i|c_j;\hat{\theta_j})}{P(d_i|\hat{\theta})} ---------- (4)$$

This existing work uses the multivariate Bernoulli and multinomial model of NB. In the multivariate Bernoulli model, a document is represented with binary vector of words. Each dimension of the space corresponds to word vocabulary. Dimension of the vector for document is either 0 or 1, indicating whether the word occurs at least once in the document. In contrast, multinomial model of NB uses representation of a document as a vector of word occurrences and this information on frequency of each word can help in classification. Documents classification is an example of a domain with large number of attributes. These attributes are words, and the number of different words in a document can be large. NB has been successfully applied to document classification.

HIERARCHICAL SELF-ORGANIZING MAP (SOM)

Hierarchical Self-Organizing Map (H-SOM) is one of pattern recognition and classification algori**thms**. H-SOM is an ANN model that depends on focused learning and is an unsupervised learning worldview. Kohonen's Self-Organizing Map utilizes an arrangement of neurons for the most part in 2-D rectangular or hexagonal framework. Information decrease into 2-D dimensionality from high dimensionality is powerful in est**imation** of comparability relations. Furthermore, **it is** valuable for information representation to help in deciding classes or comparative patterns.

H-SOM moves the subjective elements of approaching information signals into a couple of dimensional guide, and learns or finds the fundamental structure of the information. H-SOM has two layers of neurons **namel**y, an input layer and **an** output layer. Each input vector is completely associated with every neuron in the output layer. The decision of the H-SOM network size is controlled by the level of degree of the discoveries; more generalization of finding requires less matrix measure than increasingly one.

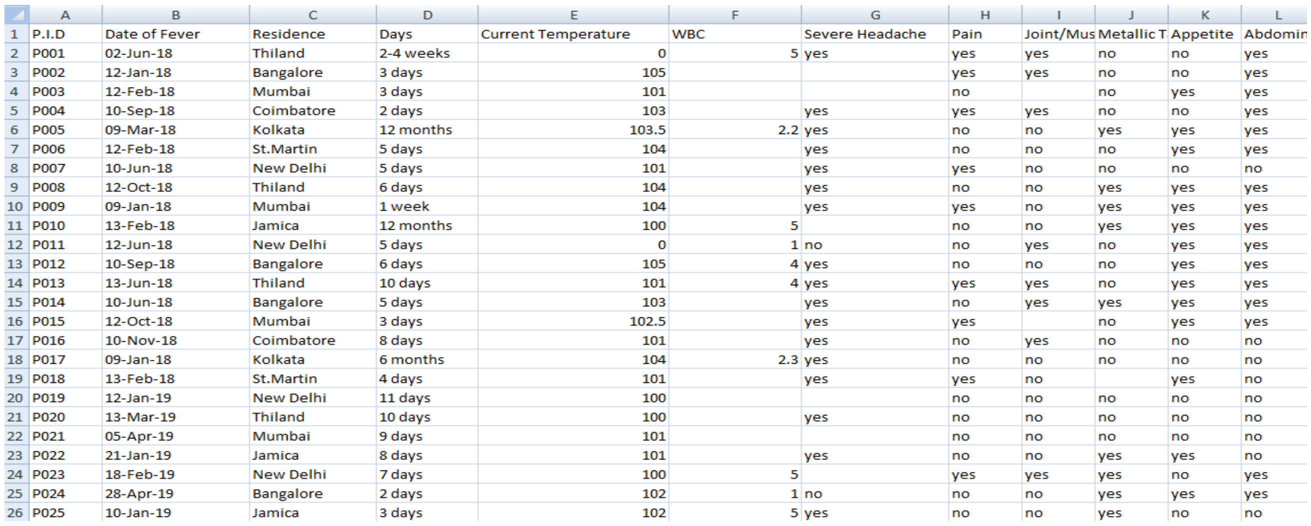
In H-SOM grid, the comparable clusters are neighbors, and various groups are long way from one another on the framework. This is called spatial autocorrelation. In the field of software building, H-SOM has been utilized in numerous inquiries and however, few are concerned about to distinguishing metric based reusable parts. Apply H-SOM to the description of the product segments, so as to classify these parts in stores that got from programming manual and the outcomes were promising .The proposed methodology is helpful for fashioner to analyze and shape a profound downplaying of the bunches to perceive the module measure. H-SOM algorithm is computationally the most straightforward and the lightest, and it is produced by and by helpful consequences of mapping a high dimensional input vector.

//*ALGORITHM*//

1. Choose random values for the initial weight vectors W_j.
2. Select a sample training input vector x randomly from the input space.
3. Find the best winning neuron $I(x)$ that has weight vector closest to the input vector that is minimum value of $d_j\,(x) = \sum_{i=1}^{D}(x_i - w_{ji})^2$.
4. Apply the weight update equation $\Delta w_{ji} = \eta(t)T_{j,I(X)}(T)(x_i - w_{ji})$ where $T_{j,I(X)}(t)$ is a Gaussian neighbourhood and $\eta(t)$ is the learning rate
5. Goto to Step 1-4 until the decrease the size of the neighbourhood. Repeat until weights are stabilized.

4. EXPERIMENTAL RESULTS

Dataset is a collection of data or a single statistical data, where every attribute of data represents variable and each instance has its own description. For prediction of dengue disease, we used dengue data set for prediction and classification of algorithms in order to compare their accuracy using R language. The R is programming language and software frame work for statistical analysis and graphics representation. R is an open source and a powerful programming language. Figure 1.5 shows a description of dengue dataset. The dataset used by us contains 18 attributes and 108 instances for dengue disease classification and accuracy. It shows the dengue dataset for H-SOM and Naïve Bayes.



	A	B	C	D	E	F	G	H	I	J	K	L
1	P.I.D	Date of Fever	Residence	Days	Current Temperature	WBC	Severe Headache	Pain	Joint/Mus	Metallic T	Appetite	Abdomin
2	P001	02-Jun-18	Thiland	2-4 weeks	0	5	yes	yes	yes	no	no	yes
3	P002	12-Jan-18	Bangalore	3 days	105			yes	yes	no	no	yes
4	P003	12-Feb-18	Mumbai	3 days	101			no		no	yes	yes
5	P004	10-Sep-18	Coimbatore	2 days	103		yes	yes	yes	no	no	yes
6	P005	09-Mar-18	Kolkata	12 months	103.5	2.2	yes	no	no	yes	yes	yes
7	P006	12-Feb-18	St.Martin	5 days	104		yes	no	no	no	yes	yes
8	P007	10-Jun-18	New Delhi	5 days	101		yes	yes	no	no	no	no
9	P008	12-Oct-18	Thiland	6 days	104		yes	no	no	yes	yes	yes
10	P009	09-Jan-18	Mumbai	1 week	104		yes	yes	no	yes	yes	yes
11	P010	13-Feb-18	Jamica	12 months	100	5		no	no	yes	yes	yes
12	P011	12-Jun-18	New Delhi	5 days	0	1	no	no	yes	no	yes	yes
13	P012	10-Sep-18	Bangalore	6 days	105	4	yes	no	no	no	yes	yes
14	P013	13-Jun-18	Thiland	10 days	101	4	yes	yes	yes	no	yes	yes
15	P014	10-Jun-18	Bangalore	5 days	103		yes	no	yes	yes	yes	yes
16	P015	12-Oct-18	Mumbai	3 days	102.5		yes	yes		no	yes	yes
17	P016	10-Nov-18	Coimbatore	8 days	101		yes	no	yes	no	no	no
18	P017	09-Jan-18	Kolkata	6 months	104	2.3	yes	no	no	no	no	no
19	P018	13-Feb-18	St.Martin	4 days	101		yes	yes	no		yes	no
20	P019	12-Jan-19	New Delhi	11 days	100			no	no	no	no	no
21	P020	13-Mar-19	Thiland	10 days	100		yes	no	no	no	no	no
22	P021	05-Apr-19	Mumbai	9 days	101			no	no	no	no	no
23	P022	21-Jan-19	Jamica	8 days	101		yes	no	no	yes	yes	no
24	P023	18-Feb-19	New Delhi	7 days	100	5		yes	yes	yes	no	yes
25	P024	28-Apr-19	Bangalore	2 days	102	1	no	no	no	yes	yes	yes
26	P025	10-Jan-19	Jamica	3 days	102	5	yes	no	no	yes	no	no

Figure 1.5 Dengue Dataset for H-SOM and Naïve Bayes

NAÏVE BAYES

Naïve Bayes is one of the algorithms that works as a probabilistic classifier of all attributes contained in data sample individually and then classifies data problems. **By** running the algorithms using Naïve Bayes, we analyze the classifier output with so many statistics to make a prediction of each instance of the dataset.

Table 1.1 Naïve Bayes Algorithm Accuracy Values

Precision	Recall	F-Measure
0.3	0.3	0.32
0.44	0.44	0.324
0.533	0.533	0.526
0.543	0.513	0.576
0.573	0.523	0.696

The output obtained by scoring of Naïve Bayes algorithm accuracy is given in Table 1.1 **which** shows the basis of precision, recall and f-measures Figure 1.8, Figure 1.9 and 1.10 **show** the precision, recall and F-Measure of **Naïve Bayes.**

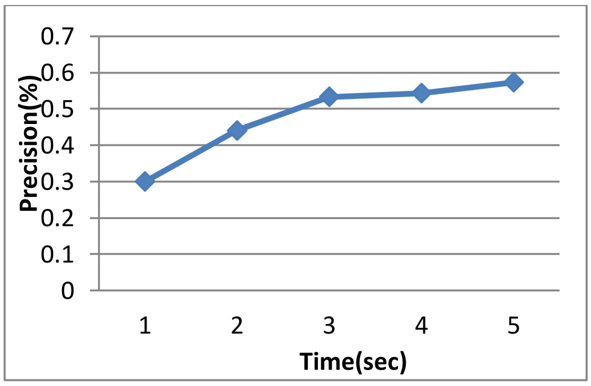


Figure 1.8 Precision Values of Naïve Bayes Algorithm

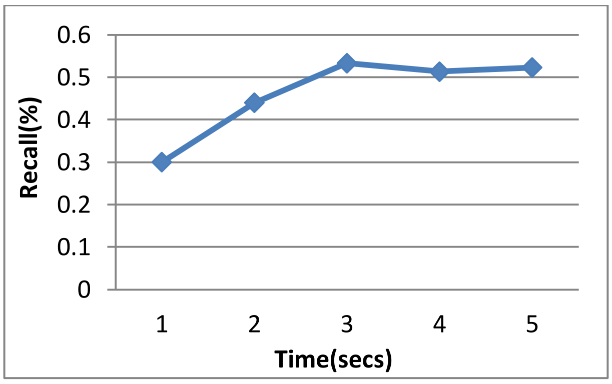


Figure 1.9 Recall Values of Naïve Bayes Algorithm

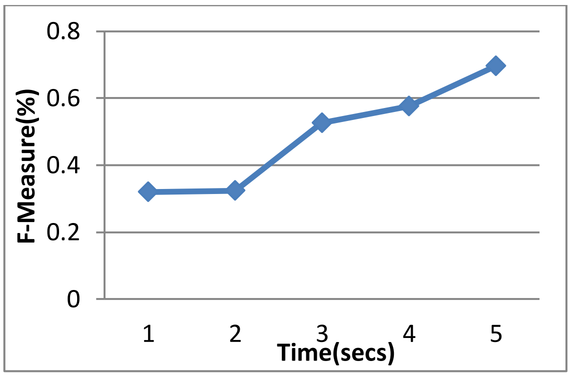


Figure 1.10 F-Measure Values of Naïve Bayes Algorithm

H-SOM

Hierarchical SOM is one of the methods used for classification. This proposed work reduces the dimensionality of the datasets without losing the information content in the dataset. This H-SOM algorithm decreases the computational time, impr**oves the** performance analysis com**pared** to the Naïve Bayes. Table 1.2 shows the H-SOM algorithm values and Figure 1.11 shows precision values, Figure 1.12 shows the recall and, Figure 1.13 shows the f-measure of H-SOM.

Table 1.2 H-SOM Classifier Values

Precision	Recall	F-Measure
0.429	0.3	0.353
0.538	0.56	0.541
0.611	0.733	0.667
0.657	0.745	0.671
0.691	0.778	0.697

PRECISION, RECALL AND F-MEASURE OF H-SOM

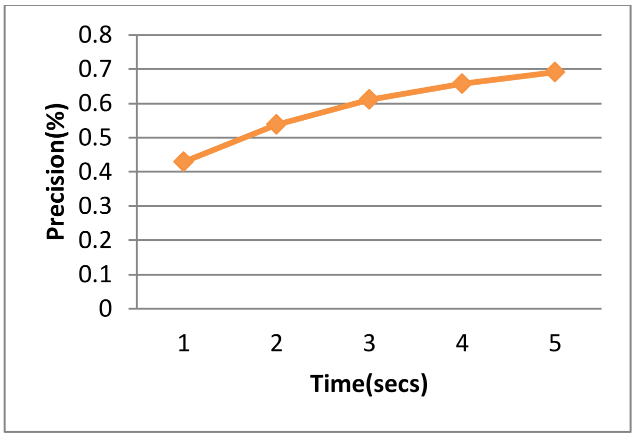


Figure 1.11 Precision Values of H-SOM Algorithm

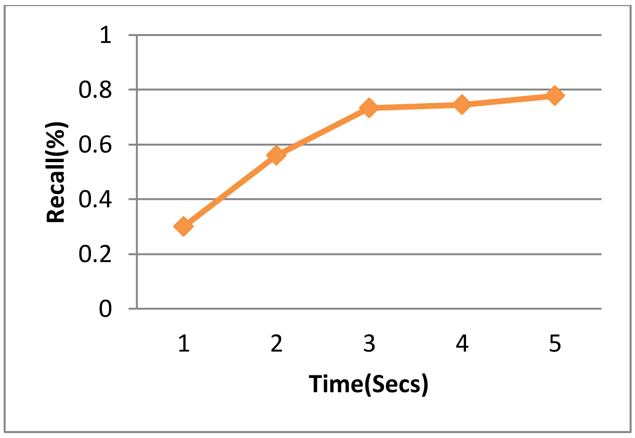


Figure 1.12 Recall Values of H-SOM Algorithm

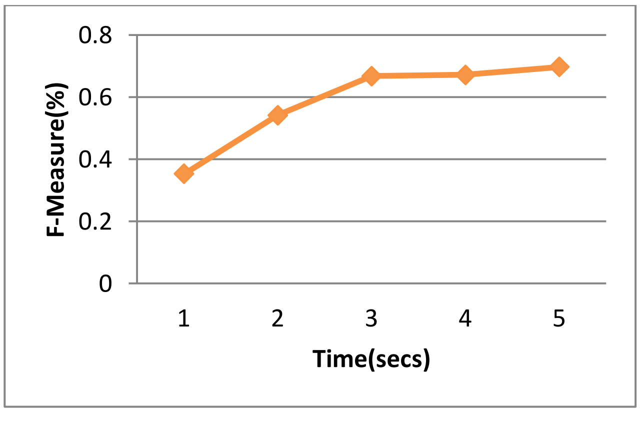


Figure 1.13 F-Measure Values of H-SOM Algorithm

COMPARATIVE ANALYSIS OF H-SOM and NAÏVE BAYES

Table 1.3 shows a comparative analysis for the existing algorithm (Naïve Bayes) and the proposed algorithm (H-SOM). Based on the performance analysis, the Naïve Bayes had a low prediction accuracy compared to the H-SOM accuracy. The results show that R data mining tool can provide accurate classification of patient's condition and improve the quality of life. Although Naïve Bayes performance was low compared to the H-SOM, the CPU time it **has taken** to produce the output results was less compared to other network.

Table 1.3 Comparisons Analysis of Naïve Bayes and H-SOM

Algorithm	Efficiency (%)	Time Taken(Secs)
Naïve Bayes	69.23	41
H-SOM	71.01	40

5. CONCLUSION

The existing detection of breast cancer cells can be predicted accurately by the use of machine learning techniques. This may result in the decrease of health cost and may enhance time required for a patient to receive treatment. In this paper, we proposed the H-SOM algorithm to provide the efficient prediction time and automatic classification problems. The R data mining tool is used to analyze algorithm accuracy which was obtained after running H-SOM and Naïve Bayes algorithms in the output window. The outputs were compared on the basis of accuracy achieved. These algorithms compare classifier accuracy to each other on the basis of correctly classified instances, time taken to build model and F-Measure. The aim of this paper **is** to apply R language in order to determine which among the lot is best for disease detection. The results showed that H-SOM algorithm is the **more** time efficient than Naïve Bayes.

REFERENCES

1. Kashish Ara Shakil, Shadma Anis, Mansaf Alam "Dengue Disease Prediction Using Weka Data Mining Tool", (2015).
2. Stefan R, Dennis Wegener, and Philipp Bremer "Re-using Data Mining Workflows",(2010).
3. Nahid Hajizadeh, Manijeh Keshtgari , Marzieh Ahmadzadeh "Assessment of Classification Techniques on Predicting Success or Failure Of Software Reusability",(2014).
4. Amjad Hudaib, Ammar Huneiti, Islam Othman "Software Reusability Classification and Predication Using Self-Organizing Map (SOM)", (2016).
5. B V Ajay Prakasha, D V Ashokaa, V N Manjunath Aradhya "Application of Data Mining Techniques for Software Reuse Process", International Journal of advanced studies in Computer Science and Engineering IJASCSE, Volume 3, Issue 8, (2014).
6. T.A.Ashok Kumar, Dr. Antony Selvadoss Thanamani "Techniques to Enhance Productivity in Manufacturing Environments Using Data Mining", (2013).
7. Divanshi Priyadarshni Wangoo and Archana Singh "A Classification based Predictive Cost Model for Measuring Reusability Level of Open Source Software", (2018).

8. Nadhem Sultan Ali, Dr. V.P. Pawar “The Use Of Data Mining Techniques For Improving Software Reliability”, (2013).
9. Priyanka Kakkar, Meenakshi Sharma, Parvinder Sandhu “Modeling of Reusability of Procedure based Software Components using Naive Bayes Classifier Approach”, (2012).
10. Tajinder Singh Sondhi, Shivam Ghildyal, Srishti Sabharwal, Akash Nagarkar, K Lavanya “Software reuse: A survey”, International Journal of Scientific Research Engineering & Technology (IJSRET), (2017).
11. Megha Dabhade, Shivam Suryawanshi, R Manjula “A Systematic Review of Software Reuse using Domain Engineering Paradigms”,(2016).
12. Nilesh Jagdish Vispute, Dinesh Kumar Sahu, Anil Rajput “A Survey on Naïve Bayes Algorithm for Diabetes Data Set Problems”,(2015).
13. K. Deeba and B. Amutha “Classification Algorithms of Data Mining”, Indian Journal of Science and Technology, Vol 9, Issue 39(2016).
14. Priyanka Kakkar,Meenakshi Sharma,Parvinder Sandu “Modeling of Reusability of Procedure based Software Components using Naïve Bayes Classifier Approach”, International Journal of Computer Applications, Vol 55 Issue 15(2012):pp 12-17.
15. Ahmed Mateen, Samina Kausar, Ahsan Raza Sattar “A Software Reuse Approach and Its Effect on Software Quality, an Empirical Study for the Software Industry”, (2017).
16. Kath, O., Schreiner, R., & Favaro, J. “Safety, security, and software reuse: a model-based approach”, fourth international workshop in software reuse and safety (2009).
17. Arman, N., & Hebron, P. “E-learning materials development: Applying and implementing software reuse principles and granularity levels in the small”, International Conferenceon E-Learning, E-Business, Enterprise Information Systems, & E-Government (2010).
18. Dantas, F., & Garcia, A. “Software reuse versus stability: Evaluating advanced programming techniques” In Software Engineering (SBES)(2010):pp 40-49