# Svs: Prediction Framework for Software Quality Enhancement through Data Mining Techniques

**A.R.Visagan, M.Sumathi, G.Sujatha**

***Abstract*: Software Engineering has its origins in tackling the issue of development and maintenance of quality software. Software Quality has been defined in multiple ways but the broadest definition is that quality is the extent to which the customer is satisfied with the developed software. Data mining has the prospects of being applied to multiple domains and addressing the long standing issues faced by them. It has been successfully applied to uncover solutions to complex problems that have long confronted these domains. The proposed research is a step in the direction. It will attempt to apply existing data mining algorithms to data accumulated by software organizations in an attempt to extract useful patterns that can go a long way in addressing the issue of software quality. This work proposed Spacious Virtue Suggestion (SVS) Model for analyzing code based quality in software quality model. The first layer of this model is Extraction Layer that extracts the various attributes of software code used. After the extraction of the metrics attributes are constructed as a vector is considered as the feature vector for the second layer of the SVS Model. The second layer of the SVS model is Selection Layer which employs feature selection strategy to obtain significant metrics attributes for the software quality prediction by reducing the overlapping metrics attributes from the vector... The third layer of SVS Model is Prediction Layer which predict the good class from the training set and result shows the high accuracy in the proposed system.***

***Keywords: Software Engineering, Software Quality Evaluation, Software Code Metrics, Spacious Virtue Suggestion, SVS Model, Artificial Intelligence, Metrics Selection, Machine Learning.***

## I. INTRODUCTION

Most significant thing in investigation of the worth of quality is perceptibility being developed cycle. It is the perceptibility picked up type cost of software quality investigation that adjust the Quality Assurance (QA) individuals worried to concentrate on those accomplishment that find, and legitimate the root purpose for the software code deserts. This underlying cause analysis through some other strategy permits the QA individuals to decide how the evolution procedure can be improved to anticipate significant zone reason for defects. Nature of a product quality is significant for both client and the engineer as the client needs to work with great qualitative programming and for designer for his reputation as well.

- Increasing dangerous of Software: The customer or buyer is surely on edge about the widespread software quality, explicitly its unwavering quality. This is progressively the situation as associations become increasingly dependent on their PC frameworks and programming is used increasingly more in regions which are security dangerous.
- The Imperceptibility of Software: This makes it extreme to know whether a particular task has been cultivated acceptably. The consequences of these assignments can be made unmistakable by requesting that the designers produce "expectations" that can be inspected for quality.
- Accumulating Errors all through programming Development: As computer framework improvement is shaped of variety of steps any place the outcome from one stage is the contribution to resultant, the faults inside the prior expectations will be added to those in the later advances bringing about a collecting delaying impact, and generally, the later in a task that an error is discovered the more costly it will be to fix. .

Generally the standard is portrayed inside the model that shows the quality attributes and relationship among them. The models assume an extremely basic job as they show what people depend on quality. The product quality models are utilized to portray an extra affixed and quantitative quality structure.

## II. REVIEW OF LITERATURE

Quality of the Software verification characterizes the guidelines (built up by Govt. or on the other hand any association) for the products created in its hierarchical unit. SQA indicate and implement tools and helps for surveying software quality [1]. Software estimating is worried about inference a numeric incentive for an aspect of a product or technique. Metric might be a quantitative proportion of how much a framework part or procedure has a given quality. It is quantitative estimating of programming, strategy or responsibility that is reasonably inspected, determined or anticipated. Measurements are marker. They are processed from measures [2]. Measurements structure the reason for arranging, investigation and controlling. They are useful to monitor necessities, anticipate improvement assets, track

**Revised Manuscript Received on December 22, 2019.**

***** Correspondence Author

**A.R.Visagan***, Department of Computer Science, Madurai Kamraj Universty College, Madurai, India.

**Dr.M.Sumathi**, Department of Computer Science, Sri Meenakshi Government Arts College for Women, Madurai, India.

**Dr,G,Sujatha**, Department of Computer Science, Sri Meenakshi Government Arts College for Women, Madurai, India.

development and comprehend conservation costs.

Chidamber and Kemerer (CK) recommend a metric set precisely for software systems that are recognized utilizing the object-oriented programming paradigm [3]. These metrics compute the degree of coupling and cohesion among the classes of a SW system, as well as the depth of inheritance relationships. The approach [4] presents survey for an Automated Software Testing using Data Mining that reviews the various approaches for software quality prediction model. Trendowicz and Punter [5] have done a remarkable audit of different systems of demonstrating quality. A software product quality model was valuable instruments toward accomplishing the objects of a software quality confirmation activity. A software product quality estimation model permits the product improvement group to follow and identify potential programming absconds moderately right on time during the advancement which is basic to numerous high confirmation frameworks.

George, Robert and Tammy [7] describes available a lot of measurements during the testing of NASA's MCCU. Their use of the measurement set during the testing exertion prompts perceives dangers and troubles right off in the test procedure, diminishing the effect of issues. Besides, by having a measurement set with great inclusion, supervisors were given more recognition into the beginnings of issues, improving the proficiency of reaction. The Booch technique [8] clarifies the investigation and configuration periods of an Object-Oriented System usage. This strategy shows a course from pre-requirements to execution by using Object-Oriented Analysis and Design and stays upon the contrast between coherent view and physical perspective on a framework.

Akintola et al. [9] played out an analysis of classifiers dependent on FFS on SDP and their outcomes offered credit to the use of FFS, yet there can even now be further examination utilizing different FS strategies. It has been demonstrated exactly that wrappers acquire subsets with preferable execution over channel include determination in light of the fact that the subsets were assessed utilizing a genuine modeling algorithm [10, 11]. Rodriguez et al. [12] have likewise focused relative analyses on FS techniques dependent on three different FFR and WRP models on four programming defect datasets. Their outcomes displayed that smaller informational indexes for the most part protect consistency with less highlights than the first informational collections.

## III. PROPOSED SCHEME

The main contribution of this proposed model is to estimating the software quality which includes three main layers. Note that when assessing the various methodologies employed in this proposal, the work pursues an exact methodology which expects access to software code metrics information. Ideally, the work might want to evaluate every strategy on both, information from open source frameworks [15]. The work mostly focuses on the quality properties of the product quality expectation. The primary explanation is that it restructures the model structure to focus on one perspective as opposed to attempting to illuminate "everything", i.e., all attributes of value, in one SVS model. In this model, different quality feature of the software quality is considered in the each layer of the work.

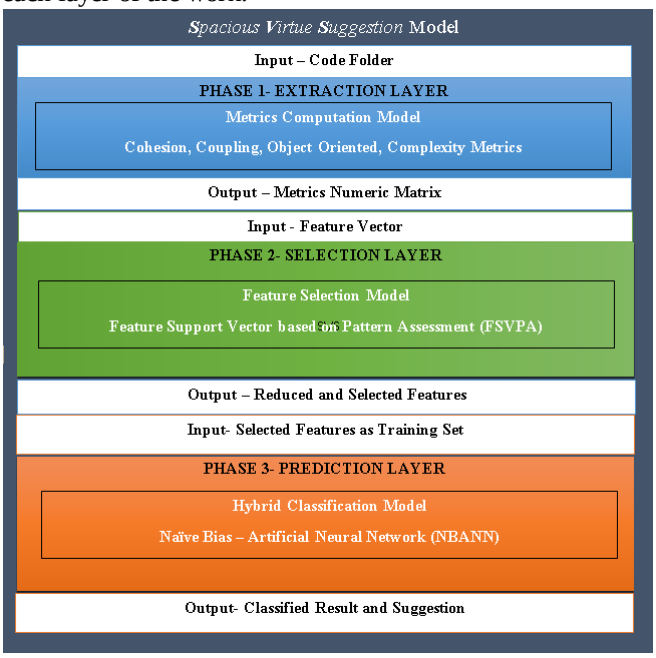


**Figure 1.1 Overall System for SVS Model**

(1) A extraction layer of the software system quality metrics from the varied quality aspects,
(2) A simplified model that is aimed toward reducing the attributes for the prediction and
(3) The significant set of input attributes and suggestions resources that for optimization, the classification strategies are utilized for the quality prediction.

3.1 Extraction Layer

In this layer, the extraction of the code based quality metrics from the software project. The input software code is included into this model for the evaluation. Then the software code is analyzed to extract the metrics. After the analysis of the code with the parser, the software quality assessment attributes are extracted in this layer [13].

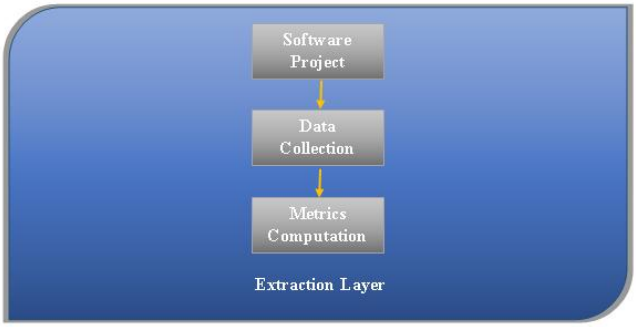


**Figure 1.2: Extraction Layer of the SVS Model**

### *3.1.1 Object Oriented Metrics*

The metrics for object oriented design concentrate on measurements that are applied to the class and method characteristics. These measurements allow designers to access the software early in method, making changes which

will reduce complexness and improve the continuing capability of the design. This metrics includes sub divisions like Package Level Metrics, class Level Metrics, and method Level Metrics.

***3.1.2Cohesion Metrics***

Cohesion illustrates the relationship of various attributes among a component. It demonstrates strength of the component. Highly cohesive component can be reused because it is an independent component. The reusability issue is increased if the component is extremely cohesive. This metrics includes sub divisions such as Tight Class Cohesion, Loose Class Cohesion, LCOM, and Method-Method through Attributes Cohesion.

***3.1.3Coupling Metrics***

Coupling is an internal attribute of the software which may be applied to be a sign of the degree of system interdependency among the components of software. Coupling is deliberated to be one in all the essential goals in software construction, which is able to ultimately result in higher maintainable, reusable and reliable software products. This metrics includes sub divisions like Coupling between objects, Friend Coupling, Inheritance Coupling, component coupling, afferent Couplings, and efferent Couplings [13].

***3.1.4 Complexity Metrics:***

Software complexity could be a regular byproduct of the functional complexity that the code is endeavoring to enable. With multiple system interfaces and complicated needs,
the complexity of software systems sometimes grows out of hand, rendering applications and portfolios too costly to preserve and risky to enhance. This metrics includes sub divisions such as Cyclomatic Complexity, Halstead Complexity, and NPATH Complexity.

**3.2 Selection Layer**

This layer includes the vector representation of the extracted attributes from the extraction layer. The vector is constructed for the quality attributes for the further evaluation. It is important to know that why feature extraction is needed in the software prediction model and modern data analysis [14].

- Dimensionality Reduction
- Automatic Exploratory Data Analysis
- Data Visualization

The feature selection strategies are employed to reduce the overlapping of the metrics for the suggestions.


**Figure 1.3 Selection Layer of the SVS Model**

**3.2.1 Feature Support Vector based on Pattern Assessment (FSVPA)**

The FSVPA [14] algorithm intends to diminish the quantity of model work calculations in inward search tree hubs. The technique requires the measure capacity to satisfy the consistent condition. The improved calculation depiction would be as per the following: the calculation activities to use the cohesion of example evaluation respect changes after expulsion of single highlights for future forecast of condition esteems without the requirement for their calculation. Prediction is permitted in specific situations as it were. Both the truly processed and anticipated condition esteems are treated similarly, for example for gathering of node descendants all through the tree development stage.

**Algorithm FSVPA**

**{**

**Input**: $M_P$- Number of all metric parameters
f- Required number of selected features
S – Set of Feature
Q- Queue
$C_V$ – Condition Value
$P_V$ – Prediction Value

**Output**: $F_{set}$ – Optimal Feature Set

1: Obtain the features of metric parameters $M_P$
2: Input the Required Number of Selected Features 'f'
3: Sort (Features) descendant values
4: Build Queue 'Q'
5: Select 'f' from S and the node
6: Create the root for the tree
7: Add root (Q)
8: Verify the $P_V$ of the current feature and not the last one
9: If True, then calculate the prediction with $C_V$
10: If False, then calculate the $C_V$ based on the all features corresponds to the selected feature
11: Evaluate whether the $C_V$ is lowest possible value
12: Then eradicate a 'f' from the set and update the 'Q'
13: Select a new feature (node) from the set
14: Then calculate the CV for the new node
15: If the CV is less than the limit then
16: Compute the PV for the feature
17: If CV is not less than compute CV for current node corresponds to all features
18: Update the CV with new update value
19: Finish the end set
20: Otherwise return feature set to repeat the process
21: Save the currently best feature subset Fset
22: }

**Prediction Layer**

In this layer, the selected quality attributes are evaluated with the construction of the machine learning. Then the evaluated attributes are processed with the existing software code attributes for the quality measurement and the quality suggestion of the software. Machine learning is a subfield of Data Mining in the field of computer science. At its core, it is the foundation for a set of statistical tools that

estimate complicated functions by learning from data. Machine learning can be sliced into two main methodologies, supervised and unsupervised learning. Supervised learning generally means that the program is given both input and the desired output, for example, pictures of objects with corresponding labels of what is depicted. The goal of the learning (or training) is to construct a map between those two. In contrast to supervised learning, the unsupervised learning approach does not provide the program with the correct output. Here, the goal of training is to find structure inside the given input; it is used, for example, in auto encoders.

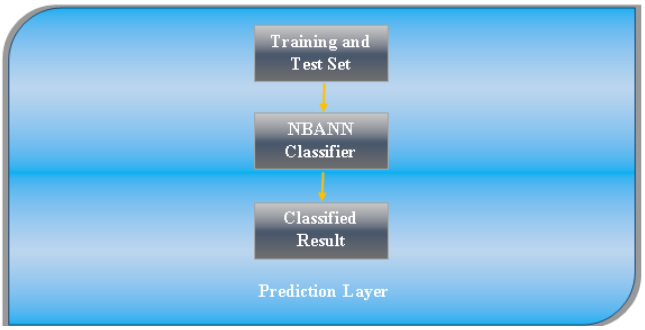


**Figure 1.3: Prediction Layer of the SVS Model**

### 3.3.1 Support Vector Machine

An SVM model is an illustration of the examples as points in space, mapped so that the examples of the separate classes are divided through a clear gap that is as extensive as possible. In addition to performing linear classification, SVMs can efficaciously perform a non-linear classification, implicitly mapping their inputs into high-dimensional feature spaces. Given a set of training examples, each marked as belonging to one or the different of two categories, an SVM training algorithm builds a model that assigns new examples to one class or the other, making it a non-probabilistic binary linear classifier. Support Vector Machines are trained so that the decision characteristic would classify the unseen example data accurately. This capacity to classify unseen example data precisely is referred to as generalization. High generalization capability is one of the important motives for the success of SVMs.

From now on, we would be searching at a two-class case (+ve class and -ve class) except specified. Let's see the fundamental notion behind the SVMs at first. Given a set of a d-dimensional vectors, a linear classifier tries to separate them with a (d−1)-dimensional hyperplane. There are many hyperplanes that might classify the data. If we define "margin" as the distance between the nearest samples on each sides of the hyperplane, SVMs are designed to select the hyperplane that has the greatest margin between the two classes. On the off chance that such a hyperplane exists, it is known as the greatest edge hyperplane and the direct classifier it characterizes is known as a most edge classifier.

**Algorithm SVM**

{

Input: Training Dataset D, Testing Set T.

Output: Output Class.

1: Read Features from the training dataset D
2: Compute α for every feature $d_i$ from D
3: Determine the support vector for every $d_i$
4: Compute b and w for the support vector
5: Evaluate the training set T and enter into the testing phase
6: Choose Input feature set FS
7: Fetch Features from FS
8: Compute α for every feature $fs_i$ from FS
9: Compute f(x)=$\sum W_i X_i + b$
10: If (f(x)>=0)
11: {
12: Classification of x Class 2
13: }
14: Else
15: {
16: Classification of x Class 2
17: }
18: Return Class

}

### 3.2.2 SVMANN

Artificial neural networks are propelled by attempts to reproduce natural neural frameworks. The human brain includes in a general sense of nerve cells named neurons, connected together with different neurons by means of remain of fiber called axons. Axons are used to transmit nerve motivating forces starting with one neuron then onto the next at whatever point the neurons are motivated. A neuron is related to the axons of different neurons by means of dendrites, which are extensions from the cell body of the neurons. The contact point between a dendrite and an axon is known as a neurotransmitter.

In this model, the blend of SVM and ANN is utilized to group the prediction framework in before stage. The hybridization strategy is utilized firstly to arrange ordinary and cardiovascular dataset utilizing direct SVM bit and afterward order the information into various stages through ANN. In SVM there will be a hyper plane between the arrangements of information focuses as the high-quality limit. For this situation, there are two grouped information of normal and abnormal information is utilized by this SVM-ANN hybrid algorithm.

**Algorithm SVM_ANN**

**{**

Input: D dataset, network, training rate

Output: a trained neural network

1: Receive the sample dataset.
2: Encode the parameters of SVM
3: Then initialize the ANN network initialization
4: Training the SVM classifier
5: Compute the weighting function for the set
6: If the condition of the lowest network weigh
7: Output optimal ANN parameters
8: Obtain the SVM identification model
9: Else

10: Compute the sum of all weights
11: Compute the new weighting function
12: If the condition met
13: Repeat the process
14: Else
15: Obtain and return optimal output class

**}**

### 3.2.3 ANNNB

The proposed methodology includes ANN and Naive Bayes (NB) for classification based on two-class of quality prediction. The progression of the proposed Hybrid methodology was additionally considered. The effective of crossover approach is by picking the best strategy for the chose individual classifiers. For instance, by considers the for all intents and purposes utilized of execution measure, work minimization/boost and the determination of threshold for ANN. Along these, this part examines in insights concerning the design and training instrument of individual classifiers that are utilized in this segment. At that point, the individual two-class classifiers (ANN and Naïve Bayes) are converged to turn into a half variety ANN-Naive Bayes classifier for programming prediction.

The outcome from the ANN becomes domain class to the essential classifier (NB). This procedure is additionally portrayed as information driven because of the fact that it utilizes the class to manage the primary classifier in constructing a result. The domain class is an additional info that is produced in another framework. Rebuilding the NB including the domain class ($d_c$) can be characterized by the accompanying condition:

$$Y_{new_n} = (Y_1, Y_2, Y_3, \dots \dots Y_n d_c = Y_k)$$

Where $y_k$ outcome of ANN. The further domain class ($d_c$) in the NB classifier is rebuilt from the first NB classifier. In the training phase, first the ANN classifier is prepared until the classifier finds the weighting for the final set and an improved choice threshold from the training data. At that point, the training information is reused to acquire $k_y$ that organizations the supposed $d_c$. The following stage requires a training process from the NB utilizing the new training set, $Y_{newn}$ Note, once more, that the $Y_{newn}$ is gathered by including the contribution from the accessible training set with $y_k$ .Finally, inconspicuous or testing information can be utilized for the prediction.

The NB accepts that each contribution to the system is permitted. The connection among data sources and result in NB organize is fixing and can't be transformed. In the Naive Bayes classifier, an estimation strategy goes about as a training technique to assess the likelihood of each class during the prediction procedure.

When all is said in done, a persistent esteemed info parameter or a Gaussian dissemination is picked to sum up the classification.

$$f(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Where x is an estimation of information $Y_i$. The mean of the information Yi is μ and $\sigma^2$ is the fluctuation of the input, $Y_i$. At that point, the possibility of all information sources Y, P ($Y_i$), given by a class, S = s, can be determined dependent on the accompanying condition,

$$P(Y) = p(S = s) \prod_{i=1}^{n} p(Y_i = y_i \,|S = s)$$

The process of prediction is computed using the probability of each input, P($Y_i$) by using the following,

$$Y^{predict} = arg\ max_s\ P(Y)$$

**Algorithm ANN_NB**

**{**

Input: Features Set F, Training Set T
Output: Output Class

1: Read Feature Set f from F
2: Weight the input feature f
3: The every input of the network is weighted
4: Random value is multiplied by the range -1 and +1
5: The weighted input is added together
6: Generate the output for ANN
7: The output of network is produced by passing that sum through the activation function
8: Train the ANN to produce the best weights
9: Compute the threshold value for the set
10: Generate the domain class using training data
11: Train the NB using new training data
12: Test the ANN to generate as domain class
13: Test the NB using new testing data.
14: Output the class

**}**

## IV. PERFORMANCE EVALUATION

This section includes the Extraction of Metrics for the Extraction Layer, Evaluation Feature Selection for the Selection Layer and Evaluation of performance for the prediction layer that employs the classification methodologies. Table 1.1 shows the sample Object oriented metrics [13] from the Project cyvis downloaded for evaluate the metrics from Github [15].

**Table 1.1 Metrics Matrix**

| cbo | wmc | dit | rfc | lcom | Private Methods | Public Method |
|---|---|---|---|---|---|---|
| 3 | 11 | 1 | 14 | 3 | 3 | 3 |
| 8 | 8 | 2 | 15 | 15 | 6 | 6 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 2 | 0 | 1 | 3 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 6 | 30 | 1 | 20 | 10 | 5 | 2 |
| 11 | 17 | 1 | 18 | 28 | 8 | 1 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 5 | 14 | 1 | 22 | 6 | 4 | 4 |
| 6 | 35 | 2 | 35 | 78 | 13 | 7 |
| 0 | 1 | 6 | 16 | 0 | 1 | 1 |
| 9 | 35 | 5 | 40 | 36 | 9 | 6 |
| 6 | 25 | 4 | 22 | 105 | 15 | 12 |
| 2 | 7 | 1 | 4 | 3 | 3 | 3 |
| 1 | 5 | 1 | 0 | 1 | 2 | 2 |
| 2 | 5 | 1 | 2 | 6 | 4 | 4 |
| 2 | 14 | 1 | 8 | 21 | 7 | 7 |
| 9 | 40 | 1 | 15 | 10 | 5 | 4 |
| 0 | 1 | 2 | 3 | 0 | 1 | 1 |
| 0 | 1 | 2 | 6 | 0 | 1 | 1 |
| 1 | 11 | 1 | 17 | 10 | 5 | 5 |
| 3 | 5 | 1 | 7 | 6 | 4 | 1 |

### 4.1 Sensitivity Ratio

The sensitivity ratio [14] evaluation for the feature selection methodologies is evaluated in Figure 1.4. The figure shows that proposed FSVPA gives high sensitivity ratio than other existing methodologies.

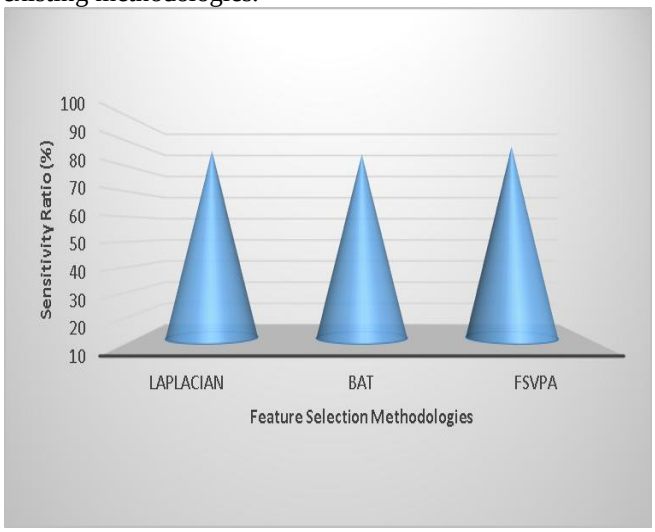


**Figure 1.4: Sensitivity Ratio for Feature selection**

| Metrics | Laplacian | BAT | FSVPA |
|---|---|---|---|
| Accuracy | 83.457 | 87.736 | 91.246 |
| Sensitivity | 87.462 | 86.094 | 89.356 |
| Specificity | 84.768 | 87.989 | 91.384 |

**Table 1.2: Performance Metrics Evaluation**

The evaluation of the performance metrics such as accuracy, sensitivity and specificity is depicted in Table 1.2. The table illustrates that proposed FSVPA provides high ratio for all performance metrics than other methods.

### 4.2 Specificity Ratio

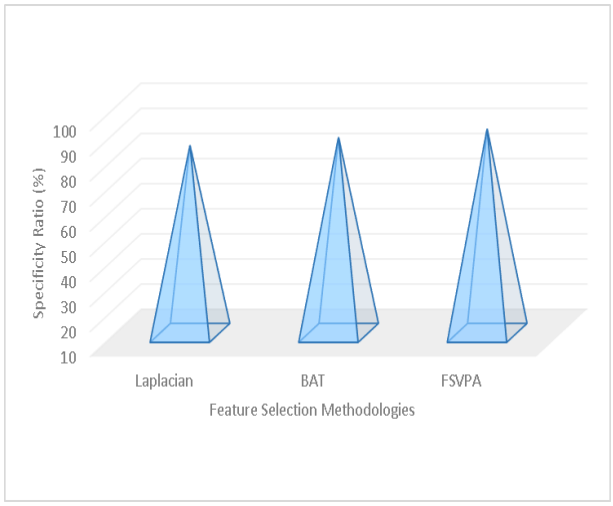


**Figure 1.5: Specificity Ratio for Feature Selection**

In the above Figure 1.5 the evaluation of specificity ratio [14] for feature selection methods is illustrated. The performance metrics evaluation for the proposed and existing classification methods as follows:

4.3 Prediction Accuracy

The Prediction accuracy is used to evaluate the performance of machine learning based system. Machine learning models have been used in the study for training by simultaneously deploying it dynamic metric suite.

$$Prediction\ Accuracy = \frac{Class_{Correct}}{Class_{All}}$$

Where the prediction accuracy for the classification methodologies is evaluated the above equation. ClassCorrect denotes to the items classified correctly while prediction and ClassAll denotes all the items classified.

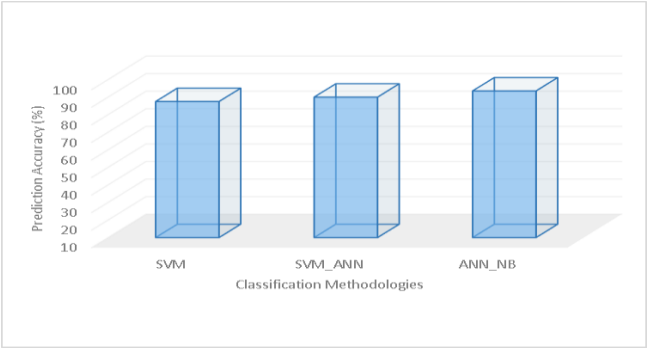


**Figure 1.6: Prediction Accuracy for Classification Methodologies**

The evaluation of prediction accuracy for proposed ANN_NB classifier is compared other classifiers is shown in Figure 1.5. The figure depicts that the proposed ANN_NB gives high prediction accuracy than other methods.

### 4.4 F1-Measure

F1 Measure is the harmonic mean of Precision and Recall. Therefore, this measure proceeds both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if it have an uneven class distribution.

$$F1\ score = 2 * \frac{precision \times recall}{precision + recall}$$

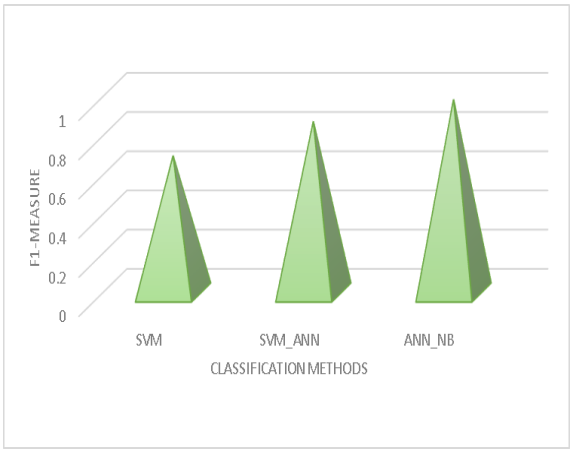


**Figure 1.7: F1-Mesure Evaluation**

The evaluation of F1-measure shown in Figure 1.6 which depicts that the proposed ANN_NB provides high harmonic value than other methods.

| Metrics/Methods | SVM | SVM_ANN | ANN_NB |
|---|---|---|---|
| Prediction Accuracy | 87.4782 | 89.985 | 93.5238 |
| F1-Measure | 0.69878 | 0.8742 | 0.9867 |

**Table 1.3: Performance Metrics Evaluation**

## V. CONCLUSION

The SVS Model for evaluate the Code Metrics that is very useful before publish the Software for the Client Use. This model has three layers first one is metric extraction layer which identifies the software code metrics. Then the software metrics are converted to identify by the machine so that it is constructed as a feature vector. Feature vector is reduced and select the best features for improve machine learning algorithm so that FSVPA evaluated in the next Selection Layer. At the end the prediction layer evaluates the classification process such as software quality prediction is better or not. Various classification methods are discussed with illustrations in the End Layer the proposed method ANN_NB is employed in the prediction of the software quality is described. Finally the work completes with the performance evaluation for the proposed and existing methodologies to measure the performance. In future various metrics are added for the most perfect prediction in this System it will continue towards in the researcher Post Doctorate.

## REFERENCES

1. J. E. Gaffney, Jr, 1981," Metrics in Software Quality Assurance," ACM '81, November 9-11.
2. Farooq Sheikh U, Quadri S M K, Ahmad, "Software Measurement and Metrics: Role in Effective Software Testing", IJEST, Vole 3, No 1, Jan 2011.
3. Chidamber S. & Kemerer C. (1994). A metrics suite for object oriented design. IEEE Transaction of software engineering, 20, 6, 476-493
4. A.R.Visagan, Dr. M.Sumathi, Dr.G.Sujatha, "A Survey on the Application of Data Mining Techniques for Software Quality Enhancement ", International Journal of Advanced Research Trends in Engineering and Technology (IJARTET), ISSN 2394-3785, Vol. 4, Issue 2, 2017 February.
5. Trendowicz, A. and Punter, T., "Quality Modeling for Software Product Lines", 7th ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering, 2003
6. A.R.Visagan, Dr. M.Sumathi, Dr.G.Sujatha, "Building a Data Mining Based Software Reliability Estimation Model ", International Conference on Advances in Information Technology.
7. George E. Stark, Robert C. Durst, Tammy M. Pelnik "An Evaluation of Software Testing metrics for NASA's Mission Control Center" 1992.
8. G. Booch, Object-oriented analysis and design, BenjaminCummings, U.S.A, pp.107-215, 1994.
9. Akintola, A.G.; Balogun, A.O.; Lafenwa-Balogun, F.B.; Mojeed, H.A. Comparative Analysis of Selected Heterogeneous Classifiers for Software Defects Prediction Using Filter-Based Feature Selection Methods. FUOYE J. Eng. Technol. 2018, 3, 134–137.
10. Lee, S.-J.; Xu, Z.; Li, T.; Yang, Y. A novel bagging C4. 5 algorithm based on wrapper feature selection for supporting wise clinical decision making. J. Biomed. Inf. 2018, 78, 144–155.
11. Zemmal, N.; Azizi, N.; Sellami, M.; Zenakhra, D.; Cheriguene, S.; Dey, N.; Ashour, A.S. Robust feature selection algorithm based on transductive SVM wrapper and genetic algorithm: application on computer-aided glaucoma classification. Int. J. Intell. Syst. Technol. Appl. 2018, 17, 310–346.
12. Rodriguez, D.; Ruiz, R.; Cuadrado-Gallego, J.; Aguilar-Ruiz, J.; Garre, M. Attribute Selection in Software Engineering Datasets for Detecting Fault Modules. In Proceedings of the 33rd EUROMICRO Conference on Software Engineering and Advanced Applications (EUROMICRO 2007), Lubeck, Germany, 28–31 August 2007; IEEE: Piscataway, NJ, USA, 2007; pp. 418–423
13. A.R.Visagan, Dr.M.Sumathi, Dr.G.Sujatha,"Refining Software Code Quality Metrics Extraction in SVS Model", International Journal of Applied Engineering Research UGC Approved, ISSN 0973-4562 Volume 14, Number 20 (2019) pp. 3841-3849.
14. A.R.Visagan, Dr. M.Sumathi, Dr.G.Sujatha, " Software metric selection in SVS model using feature selection methodologies " ,International Journal of Research and Analytical Reviews UGC Approved, E-ISSN 2348-1269, P- ISSN 2349-5138, Volume 6, Issue 2 (2019) pp.761-769.
15. [15] Github, http://github.com/.

## AUTHORS PROFILE

**A.R.Visagan.,M.Sc.,Ph.D** is an Associate Professor in the Department of Computer Science, Madurai Kamraj University College, Madurai, Tamil Nadu, India. He received his post-graduate degree in Computer Applications from Madurai Kamarj University, Karaikudi, and M.Phil. Degree in Computer Science from Bharathidhasan University, Tiruchirapalli. His research interest is Software Engineering. The author has 2 international Conference and 2 Journals. He has 17 years of teaching experience.

**Dr.M.Sumathi., M.Sc.,Ph.D** is an Associate Professor and Head in the Department of Computer Science, Sri Meenakshi Government Arts College for Women, Madurai, Tamil Nadu, India. Her research interest includes Image Processing, Pattern Recognition, Soft Computing, Biometrics, Cloud Computing and Data Mining. The author has 50 international and 5 national publications. She has 25 years of teaching experience. Six research scholars have completed their Ph.D under her guidance and 8 scholars are currently doing their research under her supervision. She has chaired International Conferences and has delivered special lectures in various Colleges and in State /National level Conferences. As an educationist she has conceptualized and implemented a new curriculum with layered learning, hands-on work and research orientation as a part of undergraduate education.

.

**Dr.G.Sujatha** received her post-graduate degree in Computer Applications from Alagappa University, Karaikudi, and M.Phil. degree in Computer Science from Mother Teresa University, Kodaikanal and Ph.D. degree in Computer Science from Madurai Kamaraj University, Madurai ,TN, South India in 2016. She is now the Associate Professor of Computer Science Department at Sri Meenakshi Government Arts College for Women (A), Madurai. She is an active researcher in Web mining, Semantic Web, Network Communication Energy Efficiency, Software Quality and Information Retrieval and has published more than 20 papers in Journals and Conference Proceedings.