**SOFTWARE DEFECT ESTIMATION USING MACHINE LEARNING ALGORITHMS**

**FINAL YEAR PROJECT**

###### ***Submitted by***

**Revoori Veeharika Reddy (9917004123)**

**Mandi Akif Hussain (9917004070)**

**NagellaKedharnath (9917004191)**

***in partial fulfilment for the award of the degree***

***of***

**BACHELOR OF TECHNOLOGY**

IN

**COMPUTER SCIENCE AND ENGINEERING**

****

**SCHOOL OF COMPUTING**

**COMPUTER SCIENCE AND ENGINEERING**

**KALASALINGAM ACADEMY OF RESEARCHANDEDUCATION**

KRISHNANKOIL 626 126

**Academic Year 2020-2021**

**DECLARATION**

We affirm that the project work titled **“SOFTWARE DEFECT ESTIMATION USING MACHINE LEARNING ALGORITHMS”** being submitted in partial fulfilment for the award of the degree of **Bachelor of Technology in Computer Science and Engineering** is the original work carried out by us. It has not formed the part of any other project work submitted for award of any degree or diploma, either in this or any other University.

Revoori Veeharika Reddy

9917004123

Mandi Akif Hussain

9917004070

Nagella Kedharnath

9917004191



**BONAFIDE CERTIFICATE**

Certified that this project report **“SOFTWARE DEFECT ESTIMATION USING MACHINE LEARNING ALGORITHMS”** is the bonafide work of “**RevooriVeeharikaReddy , Mandi Akif Hussain , NagellaKedharnath”** who carried out the project work under my supervision.

**HEAD OF THE DEPARTMENT**

Dr. A. FRANCIS SAVIOUR DEVARAJ

Professor/Head

Computer Science and Engineering

Kalasalingam Academy of Research and Education

Krishnankoil 626126

**SUPERVISOR**

S.Vidhya

Associate Professor

Computer Science and Engineering

Kalasalingam Academy of Research and Education

Krishnankoil 626126

Submitted for the Project Viva-voce examination held on.......................................

**Internal Examiner External Examiner**

**ACKNOWLEDGEMENT**

First and foremost, I wish to thank the **Almighty God** for his grace and benediction to complete this Project work successfully. I would like to convey my special thanks from the bottom of my heart to my dear **Parents** and affectionate **Family members** for their honest support for the completion of this Project work.

I express deep sense of gratitude to “Kalvivallal” Thiru. **T. Kalasalingam** B.com., Founder Chairman, “Ilayavallal” **Dr.K.Sridharan**Ph.D., Chancellor, **Dr.S.Shasi Anand**, Ph.D., Vice President (Academic), **Mr.S.ArjunKalasalingam** M.S., Vice President (Administration) , **Dr.R.Nagaraj**, Vice-Chancellor, **Dr.V.Vasudevan**Ph.D., Registrar , **Dr.P.Deepalakshmi**M.E., Ph.D., Dean (School of Computing) . And also a special thanks to **Dr.A. Francis SaviourDevaraj.** Head, Department of CSE, Kalasalingam Academy of Research and Education for granting the permission and providing necessary facilities to carry out Project work.

I would like to express my special appreciation and profound thanks to my enthusiastic Project Supervisor **S.Vidya**, Assistant Professor, CSE of Kalasalingam Academy of Research and Education [KARE] for his inspiring guidance, constant encouragement with my work during all stages. I am extremely glad that I had a chance to do my Project under my Guide, who truly practices and appreciates deep thinking. I will be forever indebted to my Guide for all the time he has spent with me in discussions. And during the most difficult times when writing this report, he gave me the moral support and the freedom I needed to move on.

Besides my Project guide, I would like to thank the rest of Class committee members and all faculty members and Non-Teaching staff for their insightful comments and encouragement. Finally, but by no means least, thanks go to all my school and college teachers, well-wishers, friends for almost unbelievable support.

****

**School of Computing**

**Department of Computer Science and Engineering**

**Project Summary**

|  |  |  |
| --- | --- | --- |
| Project Title | SOFTWARE DEFECT ESTIMATION USING MACHINE LEARNING ALGORITHMS | |
| Project Team Members (Name with Register No) | 1. RevooriVeeharika Reddy (9917004123) 2. Mandi Akif Hussain (9917004070) 3. NagellaKedharnath (9917004191) | |
| Guide Name/Designation | S.Vidya, Assistant Professor, Department of Computer Science and Engineering | |
| Program Concentration Area | Analysis of algorithms using Machine learning | |
| Technical Requirements | Python, Machine Learning | |
| **Area** | **Codes & Standards / Realistic Constraints** | **Tick ✓** |
| Economic | This project can be economic for the user as it gives them a prior informative graphical report for their analysis and choose accordingly without wasting any amount. | **✓** |
| Environmental |  |  |
| Social | It is a social project giving the service of new users getting algorithms understand well. | **✓** |
| Ethical | This sounds ethical as it gives genuine outputs and helps the users for better understanding | **✓** |
| Health and Safety |  |  |
| Manufacturability |  |  |
| Sustainability | The results of the project are very much sustainable and help it stand for the usage of users. | **✓** |

**Realistic Constraints:**

**Economic:**

The developed project in entirely economical as the coding part is done in python and it is very freely available. The datasets taken here are directly available in Google for free and are provided by NASA websites. This makes the developing page very much efficient and friendly to pocket. This doesn’t require to spend any of the amount for any loading or taking the outputs of the result as the interface is given freely for users and available for their usage than for commercial usages.

**Social:**

The developed project is very social as it binds the user with the experts to gain the most efficient algorithms and find the defects in the existing algorithms. This helps for the better understanding of the user and one can make a good use of it by opting the most efficient algorithm for his further project developed. This helps in binding socially between different people in helping each other and understanding those graphs.

**Ethical:**

This developed project is seemed to be ethical so as no bad usage of the data sets is to be done. Correct usage of the datasets and the interface results in the output of better algorithms. By getting the algorithms with minimum or no defects helps the user for better understanding and useful for the further process by better understanding. This ethical nature of the project is maintained all through the process and genuine results are given out for the users.

**Sustainability:**

This project ensures the sustainability constraints as it gives the results of the algorithm measurement in a way which is helpful for a long time and stays genuine for a longer period of time. The defects in the algorithm are shown in a graphical manner and the concept is easily understandable for the users making it to sustain for a longer period of time.

**ABSTRACT:**

Software Engineering is a branch of computer science that enables tight communication between system software and training it as per the requirement of the user. The basic uses and necessities are to be taken down initially for the client as per their usage and the following functions are to be trained for the system to make it work accordingly. It is very important all that is to be done in a limited budget with a good time and space complexities, this makes the trained software very much efficient and worthy. This mainly includes in higher performance is lesser hardware usage of the system, satisfying our clients in a very useful way. Here, our main aim is to predict the effectiveness of several machine learning algorithms and make the client select the good ones for his best usage. The effective prediction of defects is the main line project in this. By doing this, the project resources can be allocated easily with very effective algorithms in order to obtain productive outputs. This also includes the development of the system, its software, and the assurance of the quality of activities performed. We have selected seven distinct algorithms from machine learning techniques and are going to test them using the data sets acquired for NASA public promise repositories. The results of our project enable the users of this software to bag up the defects are selecting the most efficient of given algorithms in doing their further respective tasks, resulting in effective results.

**Keywords**—Software quality metrics, Software defect prediction, Software fault prediction, Machine learning algorithms

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT** | 7 |
|  | **LIST OF TABLES** | 9 |
|  | **LIST OF FIGURES** | 9 |
|  | **LIST OF ABBREVATIONS** | 9 |
| **1.** | **INTRODUCTION** |  |
|  | * 1. Problem Statement | 10 |
|  | * 1. Motivation | 10 |
|  | * 1. Objective | 11 |
|  | 1.3.1. Proposed System | 11 |
|  | 1.3.2. Advantages of Proposed system | 11 |
| **2.** | **LITERATURE SURVEY** | 12 |
| **3.** | **SYSTEM DESIGN** |  |
|  | 3.1 System Architecture | 13 |
|  | 3.2 Module description | 13 |
|  | 3.3 System Specification |  |
|  | 3.3.1 Software Requirements | 14 |
|  | 3.3.2 Hardware Requirements | 14 |
|  | 3.4 Detailed Design |  |
|  | 3.4.1 Use case Diagram | 14 |
|  | 3.4.2 Sequence Diagram | 15 |
|  | 3.4.3 Dataflow diagram | 17 |
|  | 3.4.4 Activity diagram | 17 |
| **4.** | **TEST RESULTS** | 19 |
| **5.** | **RESULTS AND DISCUSSION** | 20 |
| **6.** | **CONCLUSION AND FUTURE WORK** |  |
|  | 4.1 Conclusion | 27 |
|  | 4.2 Future Work | 27 |
| **7.** | **REFERENCES** | 28 |
| **9.** | **PLAGIARISM REPORT** | 33 |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **Table No** | **Title** | **Page No** |
| 4.1 | Test Cases | 19 |

**List of Figures**

|  |  |  |
| --- | --- | --- |
| **Figure No** | **Title** | **Page No** |
| 3.1 | System Architecture | 13 |
| 3.4.1 | Use Case Diagram | 15 |
| 3.4.2 | Sequence Diagram | 16 |
| 3.4.3 | Data Flow Diagram | 17 |
| 3.4.4 | Activity Diagram | 18 |
| 5.1 | Results (Screen shots) | 20 |

**LIST OF Abbreviations**

|  |  |
| --- | --- |
| **ACRONYM** | **EXPANSION** |
| **RBA** | Run Bagging Algorithm |
| **ML** | Machine Learning |
| **RFA** | Random Forest Algorithm |
| **MLPA** | Multi-Layer perceptron Algorithm |
| **SVM** | Support vector Machine |
| **RBF** | Radial Basis function |
| **MNB** | Multinomial Naive Bayes |

**1. INTRODUCTION**

**1.1 Problem Statement:**

Now-a-days developing software system is a difficult process which involves planning, analysing, designing, implementing, test, integrate and maintenance. A software engineer work is developing a system in time with limited budget which is done in planning phase. While doing the development process we can have few defects like not proper design, where the logic is poor, data handling is improper, etc. and these defects cause errors which lead to re-do the work, increasing in development and cost of maintenance. This all are responsible for the decrease in customer satisfaction. In this point of view, faults are grouped on the basis of sternness, corrective and advance actions are taken as per the sternness defined. The selected machine learning algorithms for comparison are

1. Bagging
2. Random Forests
3. Multilayer Perceptron
4. Radial Basis Function
5. Naive Bayes
6. Multinomial Naive Bayes
7. Support Vector Machine

**1.2 Motivation:**

On taking reference of papers the author was evaluating performance of various machine learning algorithms such as SVM, Bagging, Naive Bayes, Multinomial Naïve Bayes, RBF, Random Forest and Multilayer Perceptron Algorithms to detect bugs or defects from Software Components. Defects will occur in software components due to poor coding which may increase software development and maintenance cost and this problem leads to dis-satisfaction from customers. To detect defects from software components various techniques were developed but right now machine learning algorithms are gaining lots of popularity due to its better performance. So, by using machine learning algorithms to detect defects from software modules and by using dataset from NASA Software components and the name of those datasets are CM1 and KC1 to evaluate performance of above-mentioned algorithms.

**1.3 Objective:**

The Objective of this project is to estimate the defect of software using machine learning algorithms. On training the various ML Algorithms we need to get good accuracy percentage so that the particular algorithm fits the best in order to estimate the defects Support Vector Machine (SVM) supports both classification as well as regression. It is productive and straight-lined method which is used in classification. For classification it divides two groups by making boundaries between the group of data.

**1.3.1 Proposed System:**

The proposed system includes SVM, Multilayer Perceptron, Run Bagging algorithm, Naive Bayes algorithm, Random Forest algorithm, Multinomial NB and Radial Basis Functions to solve the class misbalancing problem which causes in the decreasing performance of defect prediction. The dataset has been trained and spitted according to the constraints and using the accuracies has been defined in order to measure the defect estimation capability of various algorithms proposed.

## 1.3.2Advantages of proposed system:

## Predicted model is used for evaluating the performance measures.

1. We can apply various datasets in this project. But we are using NASA datasets in our project.
2. Software defects are classified to the extent.

4. Advance measures can be taken on selection of algorithm

5. Provides Better results.

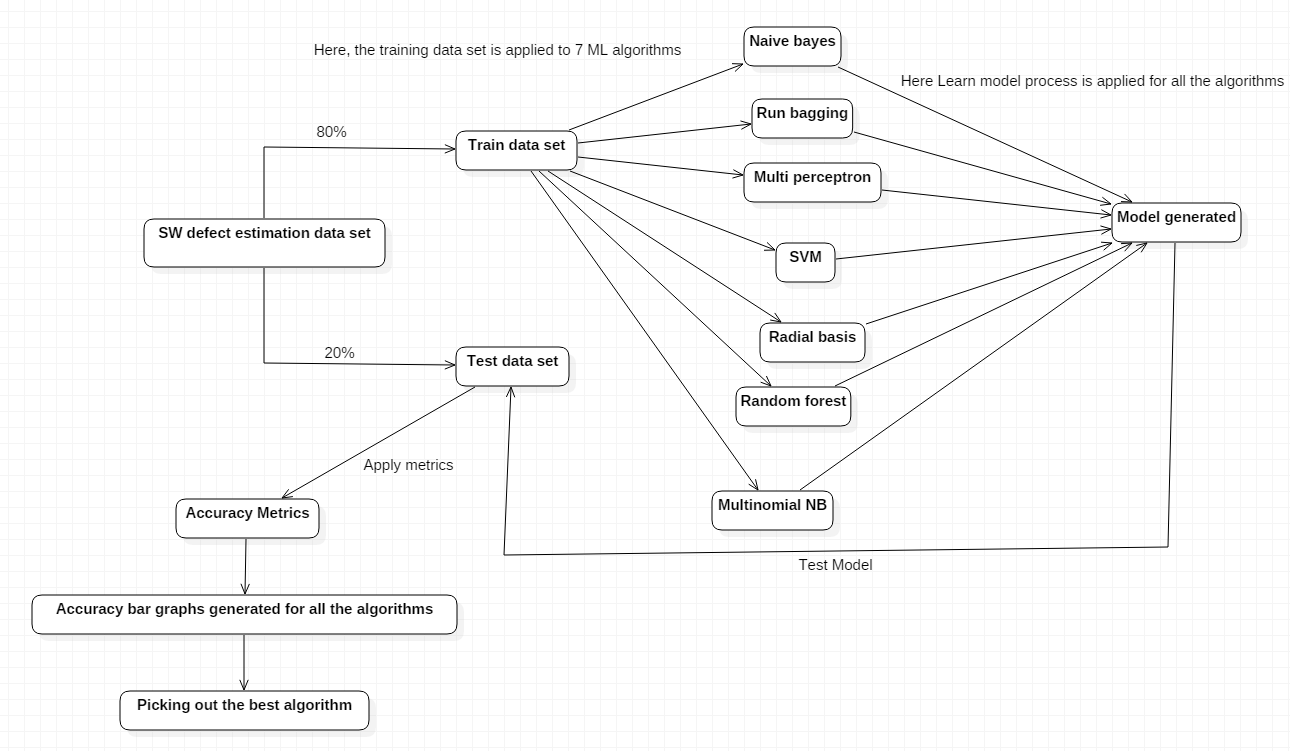
6. Identify defects in the early stage of the project which in turn results in Customer loyalty

**2. LITERATURE SURVEY**

Peng He et al. conducted a study on software defect detection with metric set. Research was conducted on 10 open sources out of 34 projects which are available at promise repository. The results of top-k metrics or minimum metric subset provides an output compared with benchmark predictors. The simplified or minimum metric set works well in case of minimum resources. Most of the present studies that are existing now on software defect prediction are restricted in performing comparative analysis of all the methods of machine learning. Few of them are using methods and provide the difference between them and others just proposed a method based on machine learning techniques by enlarging them. The present model that is existing now uses just three machine learning algorithms namely Decision tree, Multilayer Perceptron, and Radial basis function in the order to recognise the effect of the model to predict flaws on various software metric datasets that are acquired from the day-to-day life projects of three big size software companies. The solutions have shown that all the machine learning algorithms had alike results.

**3. SYSTEM DESIGN**

**3.1 System Architecture: (**Tool used: Lucid chart)

****

**3.2 Module Description:**

**Interface Designing Module:**

The main page of the project is developed using TkinterGUI .

Tkinter is the standard GUI library for Python. Python which is when interacted with Tkinter that provides a speed and simple way to create GUI applications. Tkinter which provides a object-oriented interface to the toolkit Tk GUI.

Tkinter has some standards in it ,they are:

* Dimensions
* Colors
* Fonts
* Anchors
* Relief styles
* Bit maps
* Cursors

**3.3 System Specification:**

3.3.1 Software requirements:

* Platform used: Python IDE 3.7
* Coding language used: Python

3.3.2 Hardware requirements:

* Processor - Pentium–III
* Speed – 2.4GHz
* RAM - 512 MB(min)
* Floppy Drive - 1.44MB
* Key Board - Standard Keyboard
* Monitor – 15 VGAColor

**3.4 Detailed Design:**

**3.4.1 Use case Diagram:**

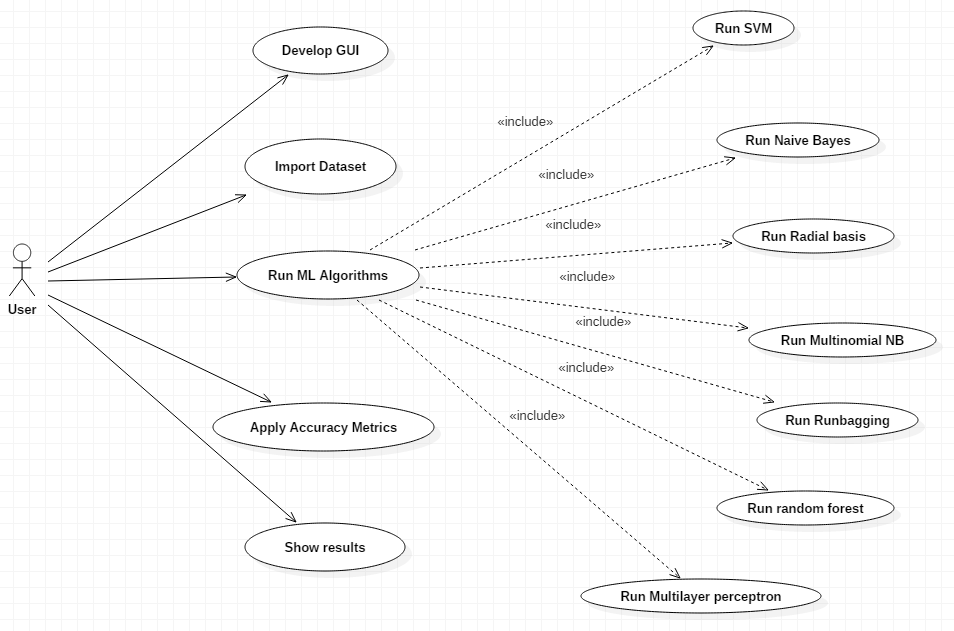
This diagram is a behavioural defined and created form analysis of use case in the language of Unified Modelling Language (UML)

The need of use case is to visualize graphical overview of the capability which is given by a system in the terms of actors, goals and any dependencies among those use cases. The main motive of the figure is to show which system functions are performed for which actor. In the system the roles of the actors can be described.

Use case diagram involves:

* Use cases
* Actors
* Dependencies

By use case diagram one can know the working of the system effectively and understand the dependencies between actor and use case.



**3.4.2 Sequence Diagram:**

It shows how the processes operate with each another and in what order. It is the construction of a Message Sequence Chart. This is explained in Unified Modelling Language (UML).

**Notations:**

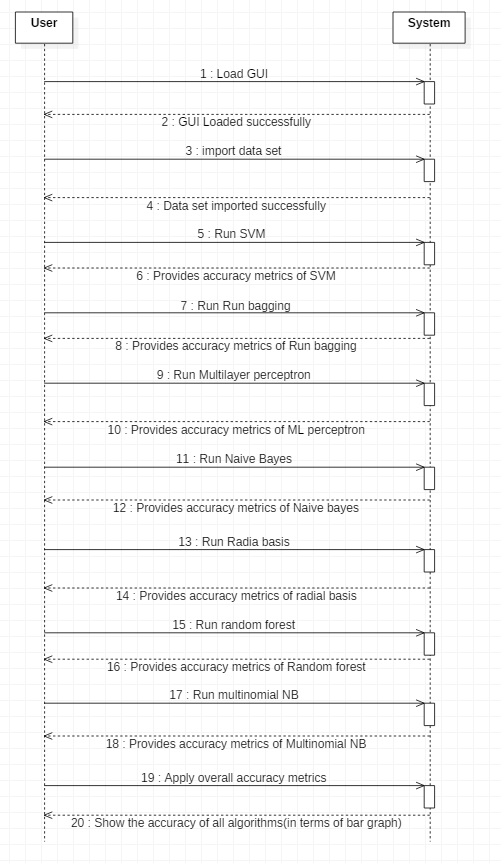
**Actors –** An actor in a UML diagram represents a type of role where it interacts with the system and its objects. It is important to note here that an actor is always outside the scope of the system we aim to model using the UML diagram.

**Lifelines –** It is named element which tells about an individual. So basically each instance in a sequence diagram is represented by a lifeline. Lifeline elements are located at the top in a sequence diagram.

**Messages –** Communication between objects is depicted using messages. The messages appear in a sequential order on the lifeline. We represent messages using arrows. Lifelines and messages form the core of a sequence diagram.

In my project there are two life lines and the message is shared between user and system

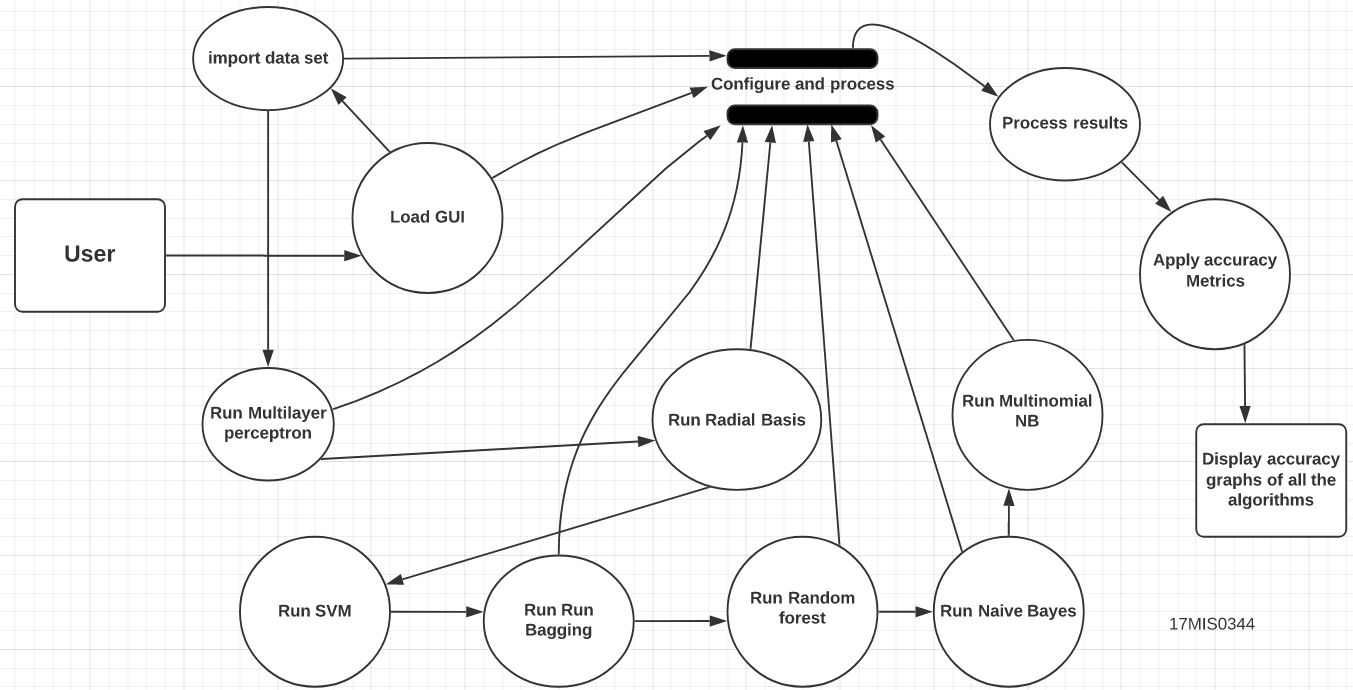
Sequence diagram goes as follows:

****

**3.4.3 Data Flow Diagram:**

* Data flow diagrams are used to represent the graph of flow of data in a business information system. DFD depicted the processes that are involved in a system to transfer data from the input to the file storage and reports generation. These diagrams can be divided into logical and physical.
* The logical data flow diagram describes flow of data through a system to perform certain functionality of a business. The physical DFD describes the implementation of the logical data flow.
* DFD graphically representing the functions, or processes, which capture, manipulate, store, and distribute data between a system and its environment and between components of a system.
* The visual representation makes it a good communication tool between User and System designer. Structure of DFD tells us starting from the overview and expands it to a hierarchy of the detailed diagrams.

**Level-1 DFD:**



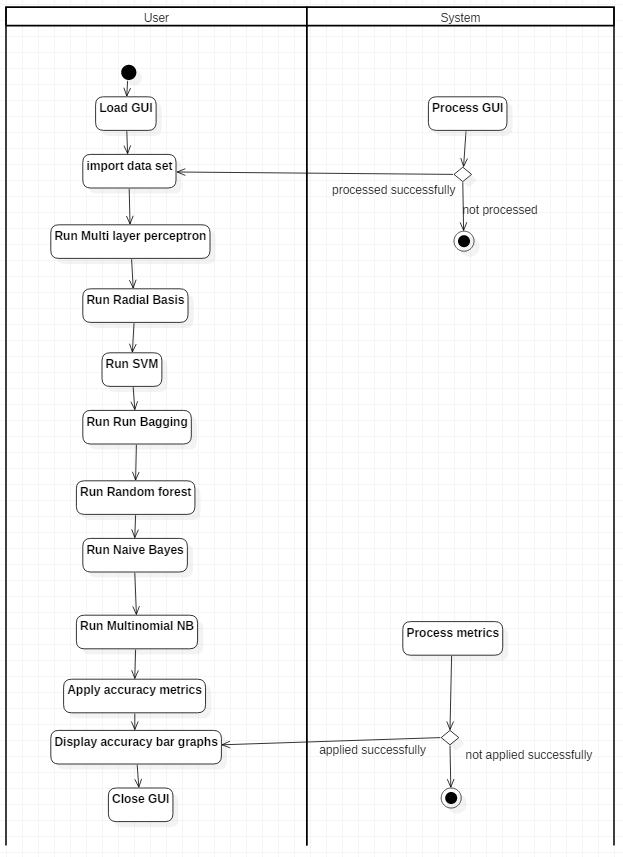
**3.4.4 Activity Diagram:**

It shows the flow of work like graphs in step wise with support for choice, loop and consistency. In the language of unified modelling, activity diagrams are used to describe the business and operational step-by-step workflow of the components in a system. The flow of control is showed here.

Benefits of activity diagrams

* Activity diagrams present a number of benefits to users. Consider creating an activity diagram to:
* Demonstrate the logic of an algorithm.
* Describe the steps performed in a UML use case.
* Illustrate a business process or workflow between users and the system.
* Simplify and improve any process by clarifying complicated use cases.
* Model software architecture elements, such as method, function, and operation.

Activity Diagram for my project is as follows: (Used Swim lane) Swimlanes are used to show which activities are performed by which organisation in the activity diagram

****

**4. TEST RESULTS**

**4.1 Test Cases:**

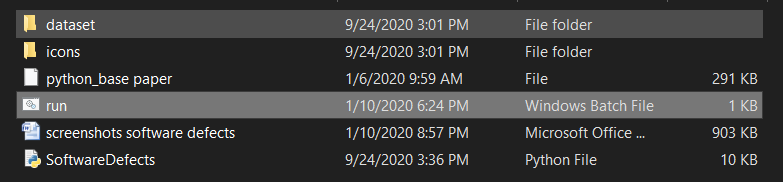
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test case ID | Input | Expected o/p | Actual o/p | Status |
| 1. | Loading GUI without clicking on the windows batch file | Display GUI | GUI cannot be loaded | Fail |
| 2. | Loading GUI on clicking the windows batch file | Display GUI | GUI displayed | Pass |
| 3. | Processing before importing the Data set | Data set imported successfully | Error | Fail |
| 4. | Processing after importing the data set | Data set imported successfully | Data set imported successfully and provides the path | Pass |
| 5. | Clicking on the “All algorithms accuracy graph” button before executing ML algorithms | Accuracy graph of all the Algorithms | Error | Fail |
| 6. | Clicking on the “All algorithms accuracy graph” after executing all ML algorithms | Accuracy graph of all the ML algorithms | Accuracy graph of all the algorithms is displayed | Pass |

**5. RESULTS AND DISCUSSIONS**

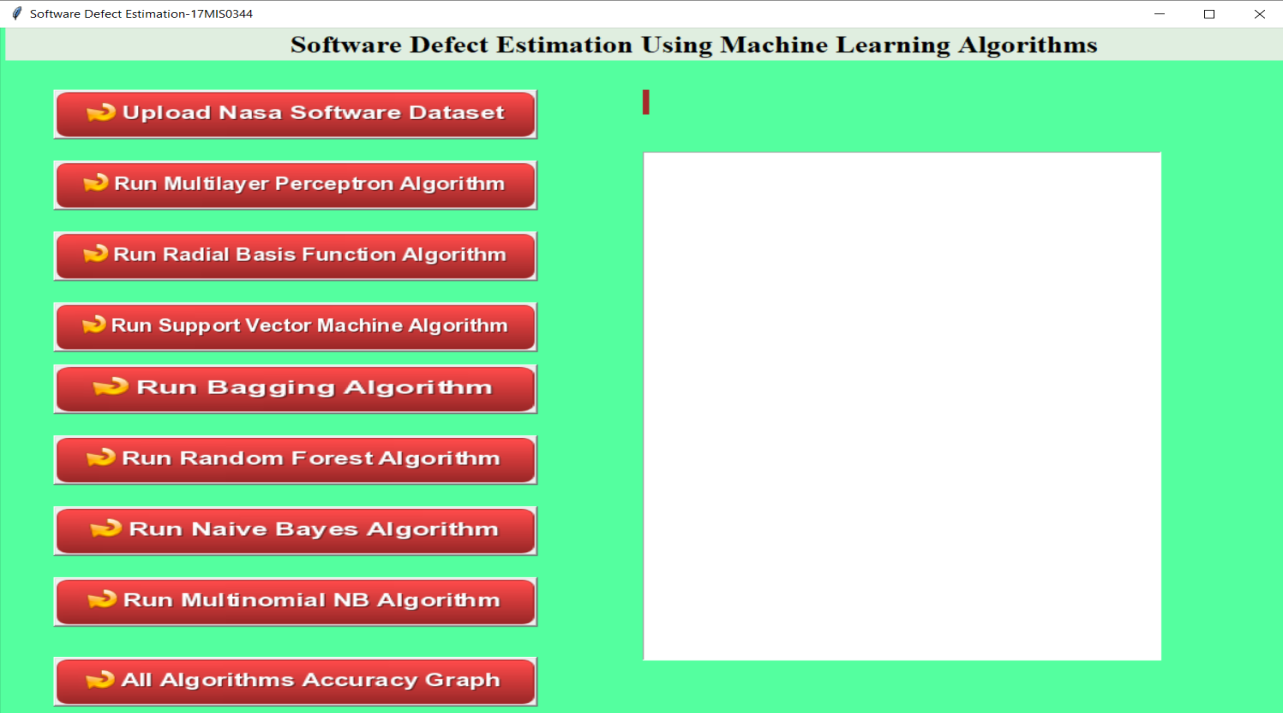
**5.1 Results:**

**Step-1:**

Double click on windows batch file to load the GUI:

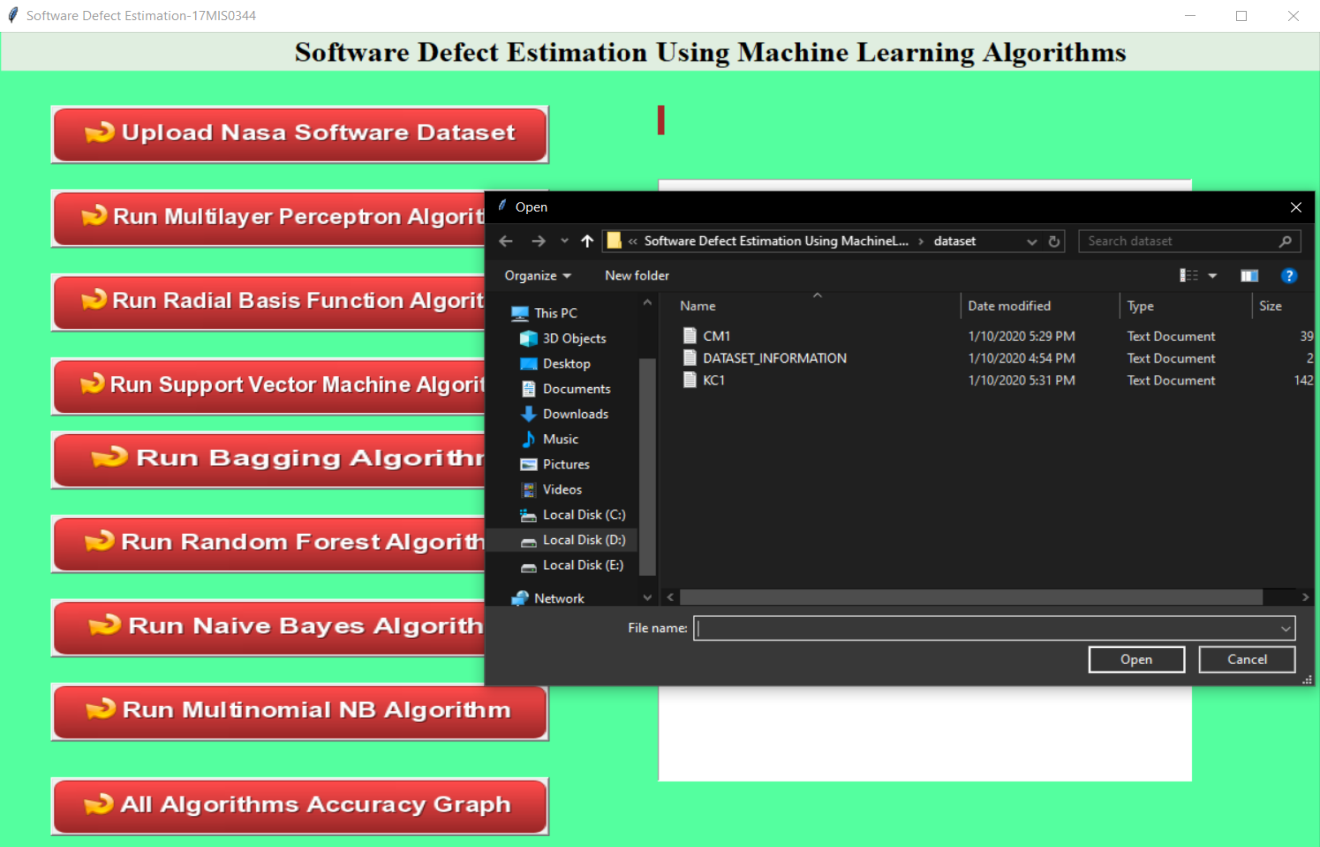


GUI:



**Step-2:**

Click on upload NASA Data Set:



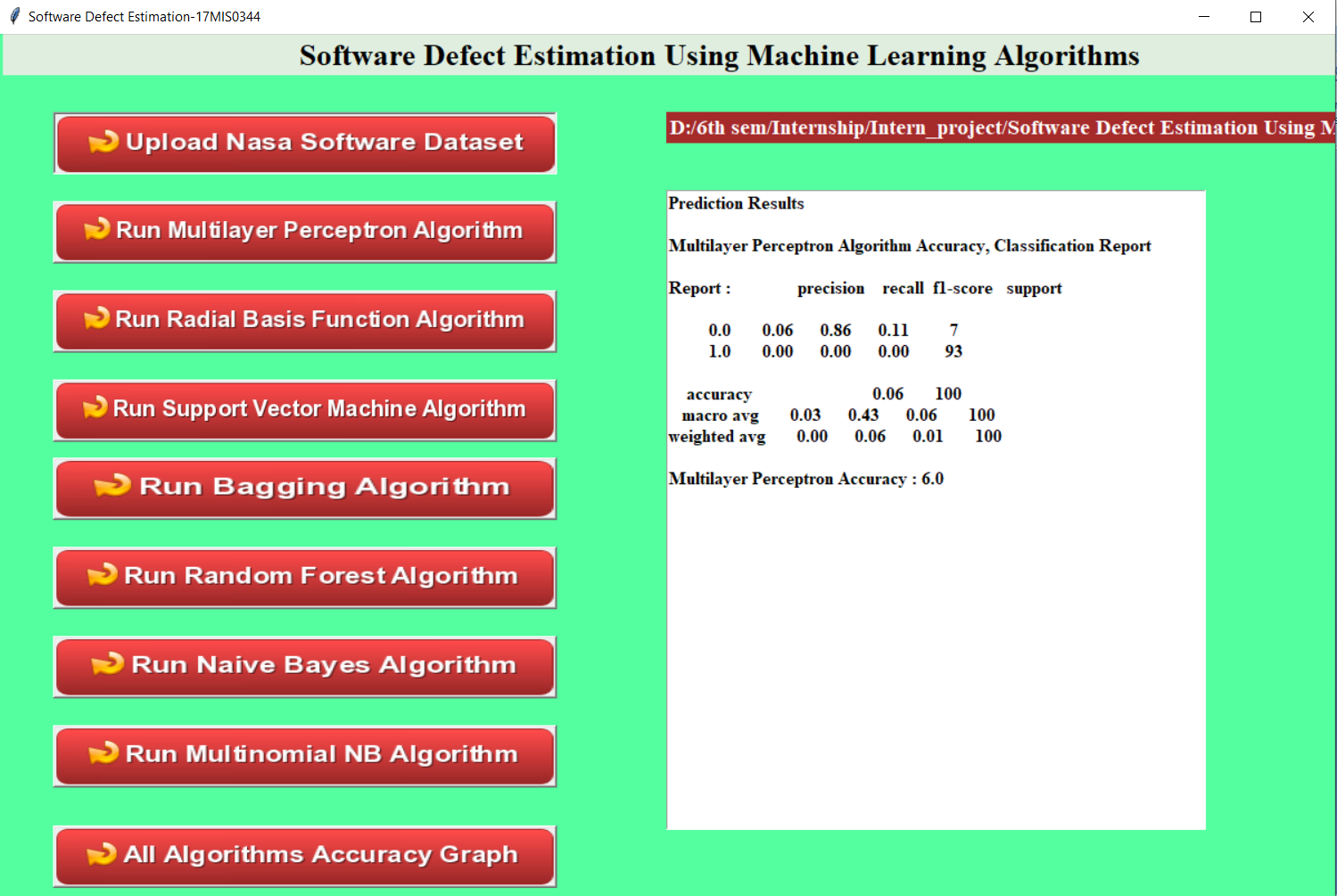
On uploading CM1 Data set:



In above screen we can see total dataset size and training size records and testing size records application obtained from dataset to build train model.

**Step-3:**

Now click on ‘Run Multilayer Perceptron Algorithm’ button to generate model and to get its accuracy

****

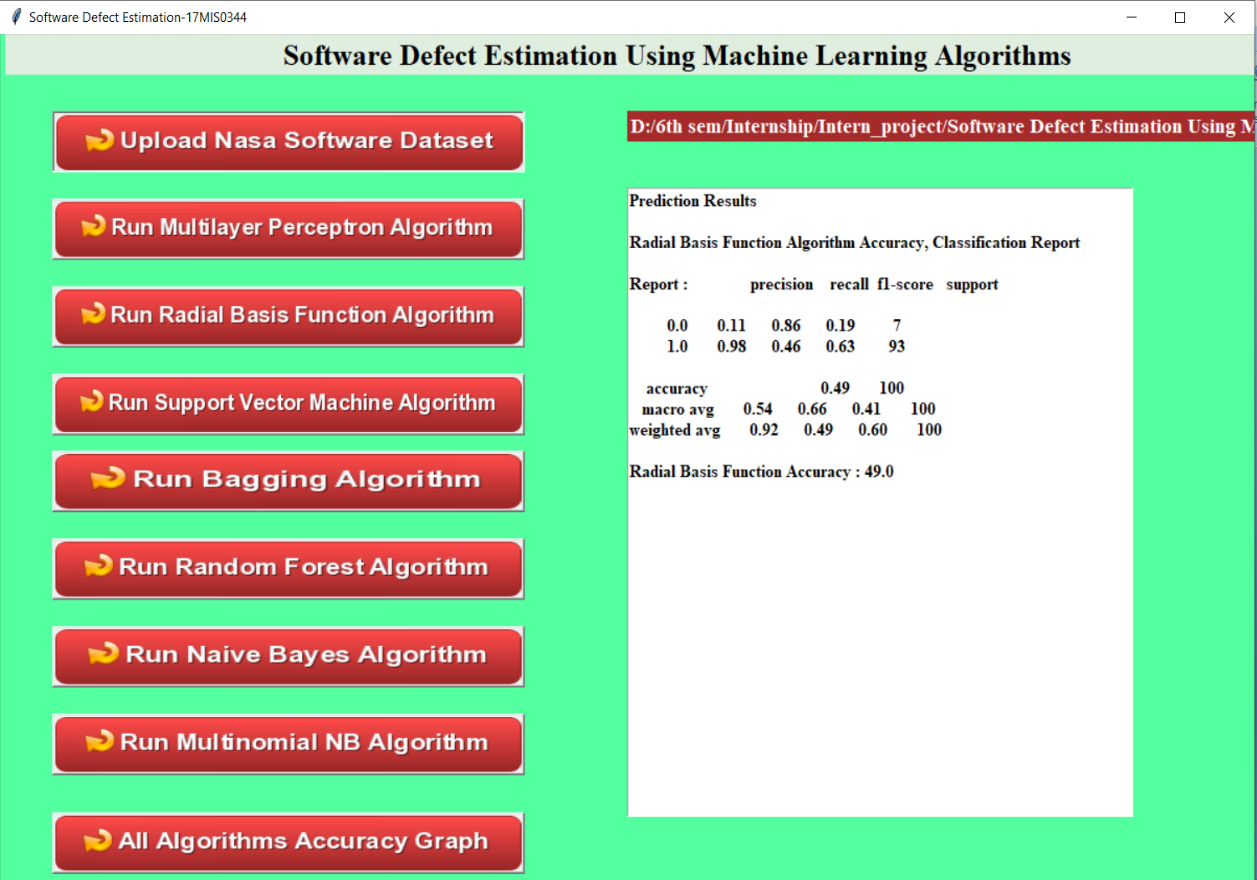
In above screen we can see multilayer perceptron f-measure, recall and accuracy values

**Step-4:**

Now click on “Run Radial Basis Function Algorithm” button to generate model and to get its accuracy

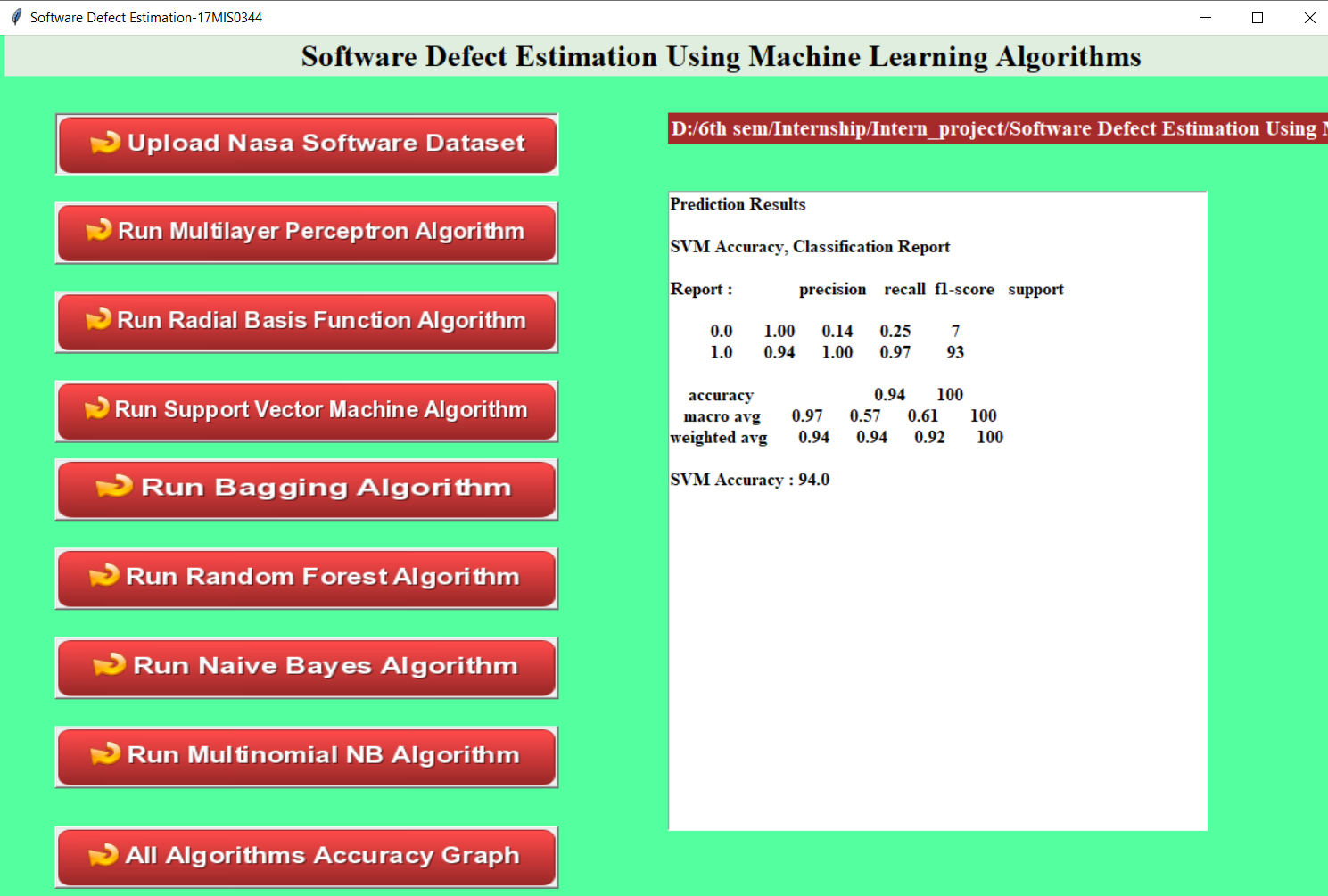
In the below screen we can see the following of the Radial Basis Function Algorithm:

* Precision
* Recall
* F1-score
* Support
* Macro average
* Weighted average

****

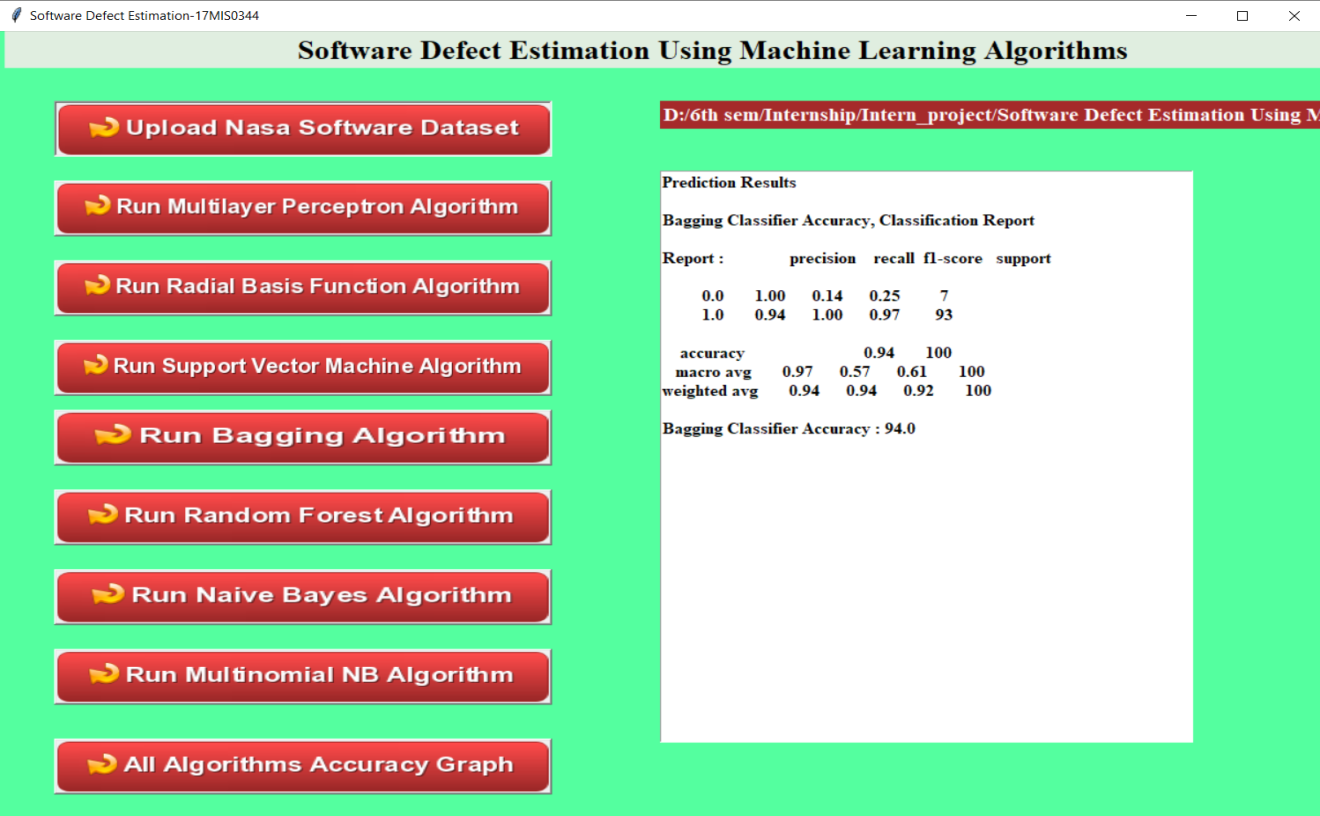
**Step-5:**

Click on Support vector Machine Algorithm:

****

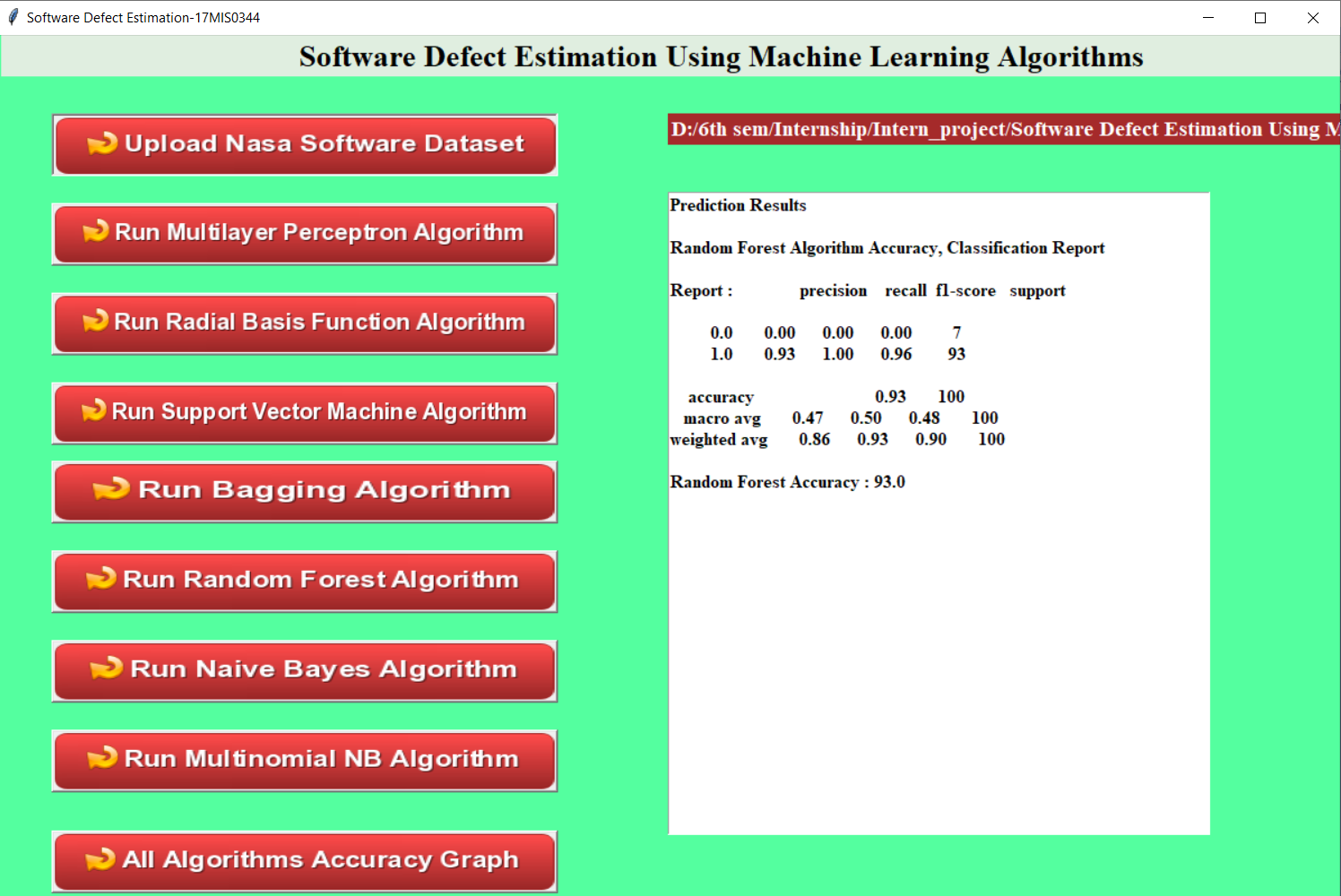
**Step-6:**

Click on Run Bagging Algorithm:

****

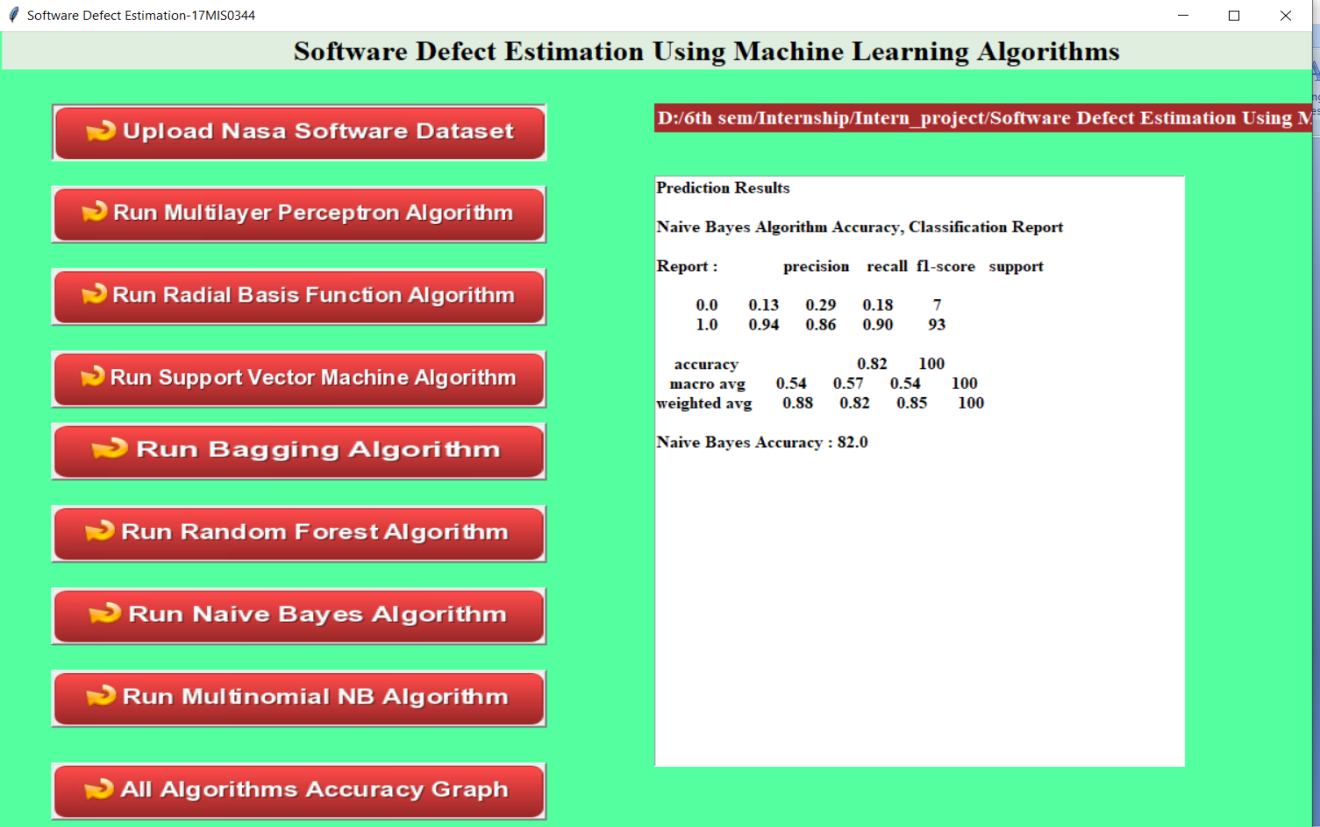
**Step-7:**

Click on Run Random Forest Algorithm:



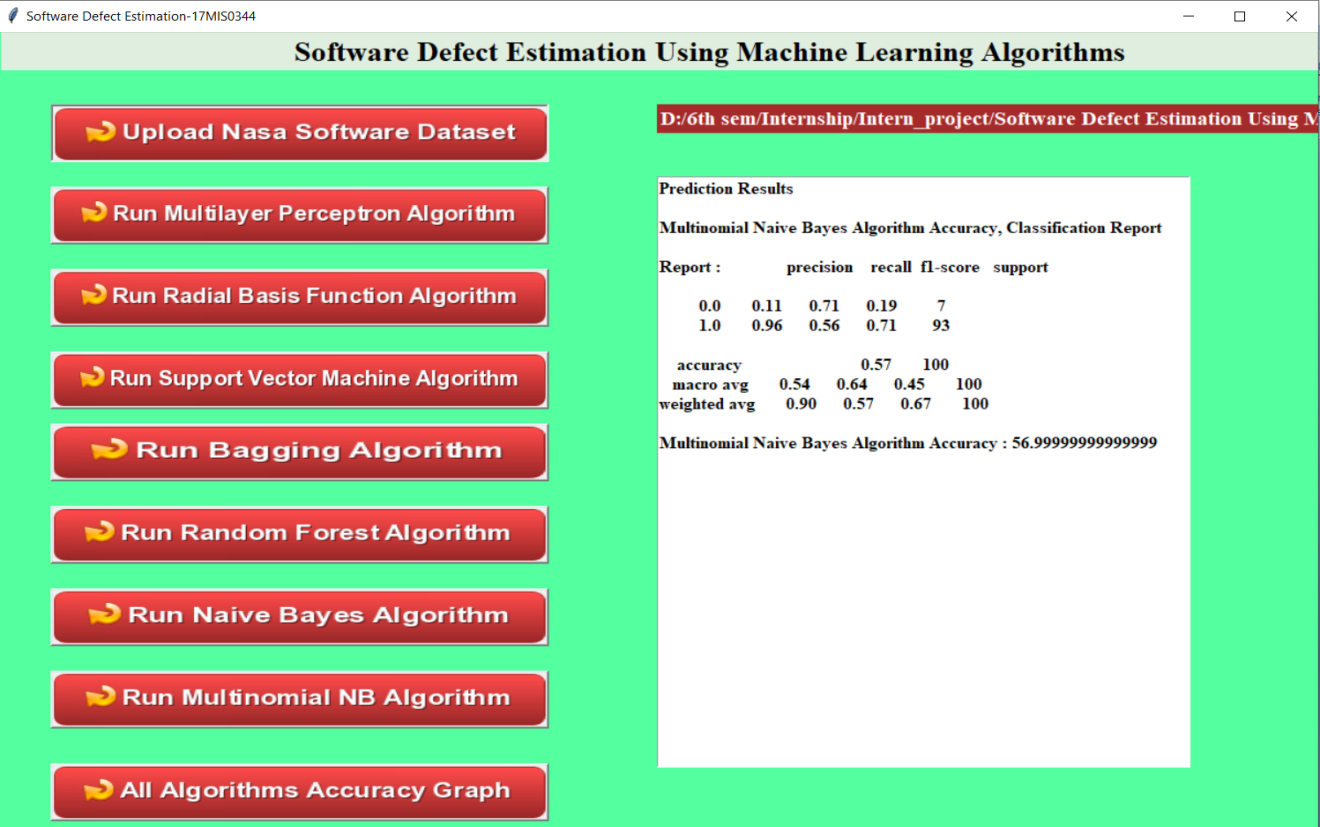
**Step-8:**

Click on Naive Bayes Algorithm:



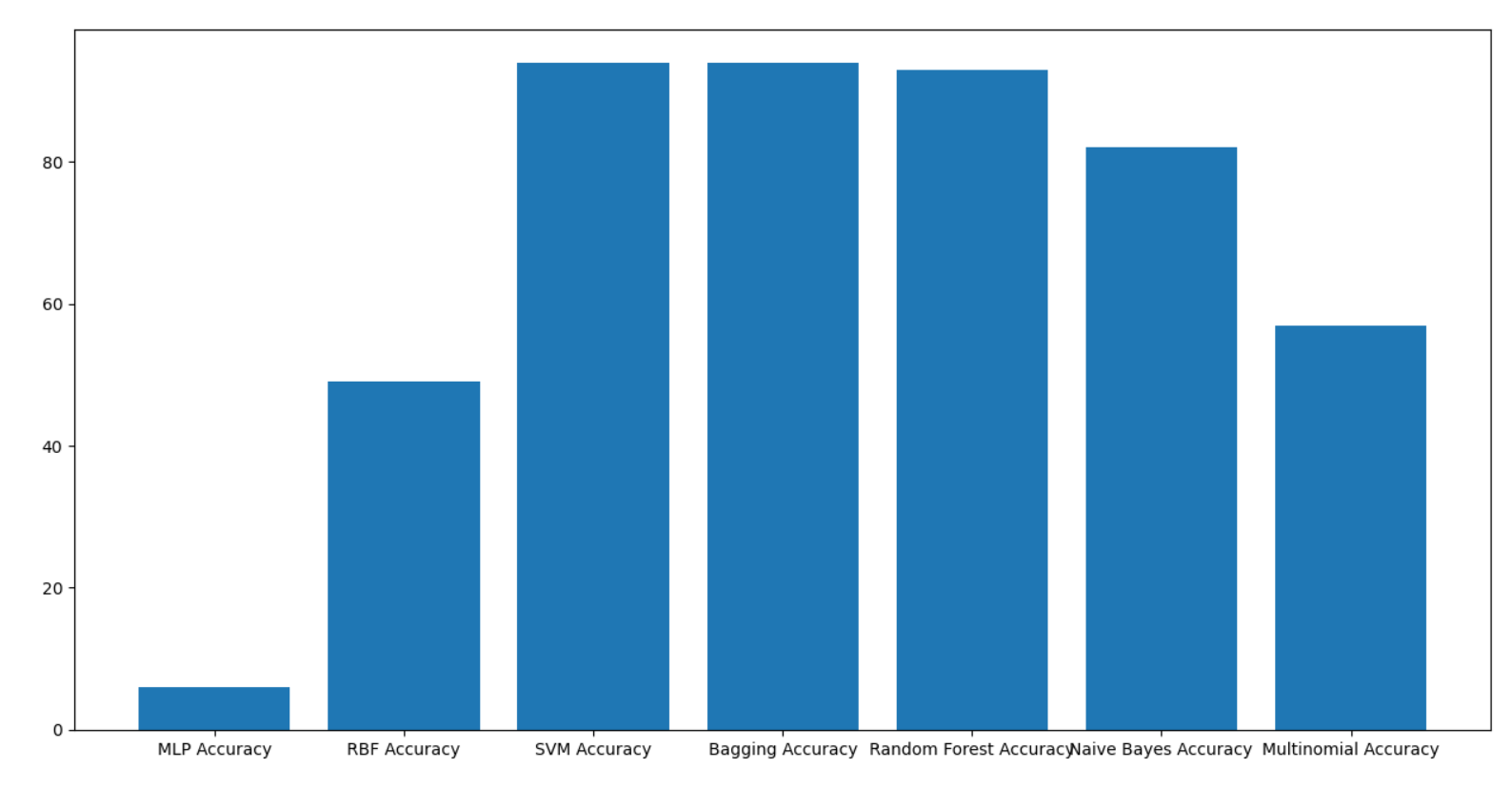
**Step-9:**

Click on Multinomial NB Algorithm:



**Step-10:**

Click on “All Algorithms Accuracy Graph” Button:



In above graph x-axis represents algorithm name and y-axis represents accuracy of those algorithms.

**5.2 Discussion:**

On seeing the above graphs, we can observe that SVM and Run bagging Algorithms produced better accuracy. So we can hereby proceed on using these two algorithms to overcome the defects of the Software and deploy a quality software to the user.In this project different evaluation metrics are used to evaluate model performance. The graph shows that the ensemble learners are better at software defect estimation and it is also a powerful way to improve the performance of the model. For each ML algorithm and corresponding data set the classification performance result is showed in the text area.

**6. CONCLUSION AND FUTURE WORK**

**6.1 Conclusion:**

Here, we are using total of seven algorithms and predict which is better algorithm in all the seven. The main algorithm which is better is used to detect the defect of the software. So, before releasing it to real environment the defects of the software are predicted by the best algorithm. The best algorithm is chosen while comparing algorithms based on the metrics of software quality which are accuracy, precision etc., the datasets I had taken for this project is CM1 and KC1 which are NASA datasets. They are taken from public repository. By using this datasets and algorithms we choose the best algorithm which saves time while software testing in early phase of cycle. So, we can take actions before they are failures.

**6.2 Future Work:**

The main aim of future work is to test it with unique data sets. These datasets are taken from organisations or open archives. Second heading of things to come work would direct a trial concentrate by applying profound learning calculations extra to these AI calculations. Bringing into reality of new credits by utilizing mix of past traits would be another course of things to come work and also testing the project with various other NASA data sets will also be a precise one in the future.

**7. REFERENCES**

[1] Biffl, S. (2000). Using inspection data for defect estimation. *IEEE Software*, *17*(6), 36-43.

[2] Petersson, H., &Wohlin, C. (1999, November). An empirical study of experience-based software defect content estimation methods. In *Proceedings 10th International Symposium on Software Reliability Engineering (Cat. No. PR00443)* (pp. 126-135). IEEE.

[3] Yalçıner, B., &Özdeş, M. (2019, September). *4th International Conference on Computer Science and Engineering* IEEE.

[4] Biffl, S. (2003). Evaluating defect estimation models with major defects. *Journal of Systems and Software*, *65*(1), 13-29.

[5] Bai, C., Cai, K. Y., & Chen, T. Y. (2003, November). An efficient defect estimation method for software defect curves. In *Proceedings 27th Annual International Computer Software and Applications Conference. COMPAC 2003* (pp. 534-539). IEEE.

[6] Boehm, B. W., & Fairley, R. E. (2000). Software estimation perspectives. *IEEE software*, *17*(6), 22-26.

[7] Fakhrahmad, S. M., & Sami, A. (2009). Effective estimation of modules' metrics in software defect prediction. In *Proceedings of the world congress on engineering* (Vol. 1).

[8] Jain, A., Patnaik, A. R., Dhar, P., & Srivastava, V. (2006, May). Software defect content estimation: A Bayesian approach. In *2006 Canadian Conference on Electrical and Computer Engineering* (pp. 2449-2454). IEEE.

**8. PLAGIARISM REPORT:**