j48 Classifier for Software Effort Estimation

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Software Engineering

Submitted By
Nancy Sharma
(Roll No. 801631014)

Under the supervision of:
Vineeta Bassi
Assistant Professor

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR INSTITUTE OF ENGINEERING AND TECHNOLOGY
PATIALA – 147004

June 2018

Certificate

I hereby certify that the work which is being presented in the thesis entitled, "*j48 Classifier for Software Effort Estimation*", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Software Engineering* submitted in Computer Science and Engineering Department of Thapar Institute of Engineering and Technology, Patiala, is an authentic record of my own work carried out under the supervision of *Ms. Vineeta Bassi* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

Signature:
(Nancy Sharma)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

(Vineeta Bassi)
Assistant Professor
CSED

Acknowledgement

This thesis has been kept on track and been seen through with the support and encouragement of numerous people including my guide, my teachers and my friends.

First of all I would like to express my deepest gratitude to my guide, **Ms. Vineeta Bassi**. This work would not have been possible without her guidance, support and encouragement. Under her guidance, I successfully overcame many difficulties and learned a lot.

I gratefully acknowledge **Dr. Maninder Singh**, Head, Computer Science and Engineering Department and **Dr. Rupali Bhardwaj**, PG Coordinator for their valuable suggestions and sharing knowledge that have been very helpful for this study. I would like to thank **Dr. S.S. Bhatia**, Dean of Academic Affairs for providing necessary infrastructure and resources to accomplish the research work.

I am thankful to the entire faculty and staff members of Computer Science and Engineering Department for their direct-indirect help, support, cooperation and love. I would also like to thank all my teachers for their guidance and for their unique way of teaching during these two years of my M.E.

Last, but not the least I deeply thank my family members for their unconditional trust and encouragement.

Nancy Sharma

(801631014)

Abstract

The software effort estimation is the technique, which is applied to estimate efforts for the software development. The software effort estimation is the challenging task due to large size of the software. The models, which are designed for the effort estimation depend upon the KLOC (Kilo Lines of Code) values. The incorrect estimation of the KLOC value leads to incorrect estimation of the effort, which raises the MRE (Mean Relative Error) value. The COCOMO (Constructive Cost Model) model is the most popular model for software's effort estimation. The effort calculated in COCOMO model is directly proportional to KLOC value. This work is based on the KLOC value estimation for the reduction of MRE. In the existing technique, random forest classifier is applied for the reduction of MRE value. In this thesis work, j48 model is applied for the reduction of MRE value. The proposed and existing techniques are implemented in Python using Anaconda software. The performance of both algorithms is compared in terms of accuracy and execution time. The j48 algorithm shows high accuracy and less execution time for the effort estimation.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

KLOC	Kilo Lines of Code
MRE	Mean Relative Error
MMRE	Mean Magnitude of Relative Error
COCOMO	Constructive Cost Model
RMU	Result Metric Unit
LOC	Lines of Code
SLOC	Source Lines of Code
PM	Person Months
EAF	Effort Adjustment Factor
ID3	Iterative Dichotomiser 3
GSE	Global Software Engineering
LMS	Least Median of Square
LQS	Least Quantile of Squares
FPA	Function Point Analysis
SLIM	Software Lifecycle Management
OSS	Open Source Software
OOB	Out of Bag
CART	Classification and Regression Tree
MSE	Mean Square Error
OLS	Ordinary Least Squares
MdMRE	Median of the Magnitude of Relative Error

Chapter 1

Introduction

The basic introduction of software project estimation is provided in this Chapter. Several tools and algorithms used within these systems are also elaborated in this chapter.

1.1 Software Project Estimation

There are many techniques introduced but it is still a challenging task to find appropriate technique, which can estimate the actual cost required to develop the software applications or systems within organizations. Initially, the high-level requirements were used to calculate the basic estimates, through which a clear view of the overall estimation cannot be achieved. There is a need to include upgraded resources within the latest technologies that are included in new projects. These resources might include latest software technologies, improved infrastructure and other improvements [1]. Thus, the estimation of actual cost of a project is almost impossible when the development of a new project is at the early stages. Size estimation, effort for development, cost of development, and schedule of development of a particular project in a specific scenario is known as software estimation. There are particular methods, tools as well as techniques used within this process. The method of predicting realistic amount of effort through which software can be developed or maintained on the basis of incomplete and uncertain input is known as software development effort estimation.

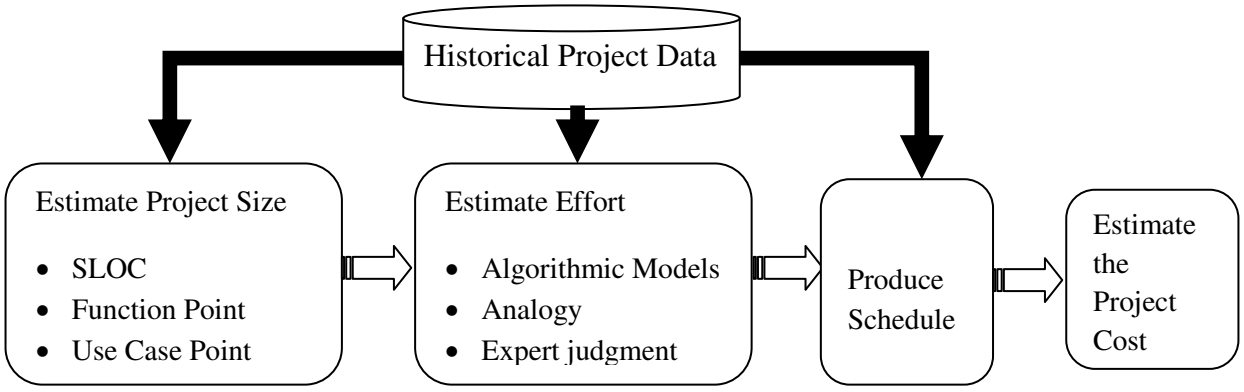

Figure 1.1 Conceptual View of Software Estimation Techniques

Some of the common steps that are followed for software project estimation are:

i. Size estimation for the development product: In order to attain effective estimate, the initial step is to estimate the size of software accurately. There should be formal descriptions of the requirements at the beginning of providing sources of information related to scope of project. For providing additional details, the design documents can be used during the (re-)estimation of a project in the next phases of the process of a project's lifecycle. However, the initial project estimation should not be stopped due to the lack of formal scope specification.

ii. In terms of person-months or person-hours, the effort is estimated: The effort estimation can be derived once the size estimation of a product is known. Only after the defining of software development lifecycle and development process such that the software can be designed, developed and tested, the software size can be converted to total project effort. The simple coding of software covers the smallest part of overall effort of project and thus it is not the only requirement needed in software development project. The identification and estimation of activities and then summing them up so that a product of estimated size can be build, is done with the help of project effort estimation.

iii. Schedule Estimation in calendar months: Determination of project schedule from the effort estimate is the third step of estimating the software development project. Mainly, this step includes the number of people that are going to work on the project, the tasks that they are going to perform, the time at which they will be initiating their tasks and time duration in which they will be completing the tasks [2]. This information needs to be laid out in a calendar schedule once it has been gathered. In order to predict the number of people required for a particular size of project and the manner in which complete work can be broken down into a schedule, the historical data from organization's previous projects could be used.

iv. Estimation of project cost in dollars (or local currency): During the estimation of total cost of project, there are several factors to be considered. These factors include telecommunications, hardware and software rentals, travelling costs for meetings, testing and training sessions etc. On the basis of the manner in which the costs are allocated, the total project cost is estimated. The individual

projects might not allocate some of the costs and by adding an overhead value to labor rates (referred to the cost of deploying employees), these costs can be handled. Only the cost of labor is estimated by a software development project manager more often, along with the adding up of any additional project costs that the organization does not count as overhead costs.

1.1.1 The Basic Estimation Process (ABC-Steps)

A genetic procedure applied to estimate the software proves that effort is described by ABC estimation strategy. The estimation of any of the quantitative parameter of any domain can be done here. There are three basic steps included within the generic estimation strategies that are explained below:

Step A: Quantification

The complexity or size of the system model that is being generated is calculated in this step. A software complexity metric such as the System Meter, which is proposed on the system model, is applied in order to compute this value. In order to express the sizes of these types of software metrics, the expression of "metric of size" is utilized[3]. Since it is not necessary that the notion of size that is adopted from non-software domains is relevant to effort, this expression must be called "effort inducing property".

Step B: Estimation

This step computes the estimation value for which the approximation function A is applied to the p predictor value. Estimation function is another name used for representing function A. Thus, the estimated value e is defined as:

$$e = A(p)$$

The resultant value achieved is measured in Result Metric Unit (RMU). It is to be ensured that the effective outcome should be closer to the value of e. Generally, an error or bias is observed here. Along with the function A, the mean approximation bias dA, which is known to be relative percentage commonly is also available with this.

Step C: Adaptation

The process is not completed after the statistically derived estimation e is achieved. It is a non-standard process model that includes activities that one has to perform. The adapting and interpreting of resultant value is the final step of the result model. The major concerns highlighted here include:

- The risks that one can and wishes to take in order to achieve an effective outcome that is higher than e value.
- The parameters that are possibly of higher probability include interference.
- In relevance of some template result model such as, the building up of software process model, the effects of omissions or additions of result model are seen.

1.1.2 Software Project Estimation Tools

The individual products or the products that are integrated within the functionality of large scale project management products are known as estimation tools. The process of estimating the size or either converting the size into effort, cost and schedule can be supported by the estimation tools. LOC counters, Function point analysis programs & other requirements capture and management applications are the tools that support the size estimation[4]. All the estimation issues cannot be resolved by using any one estimation tool. Within the estimation toolkit, these tools can be very helpful however; the efficiency of output completely depends upon the inputs given. In order to support these tools, an estimation and software development process is required for these tools.

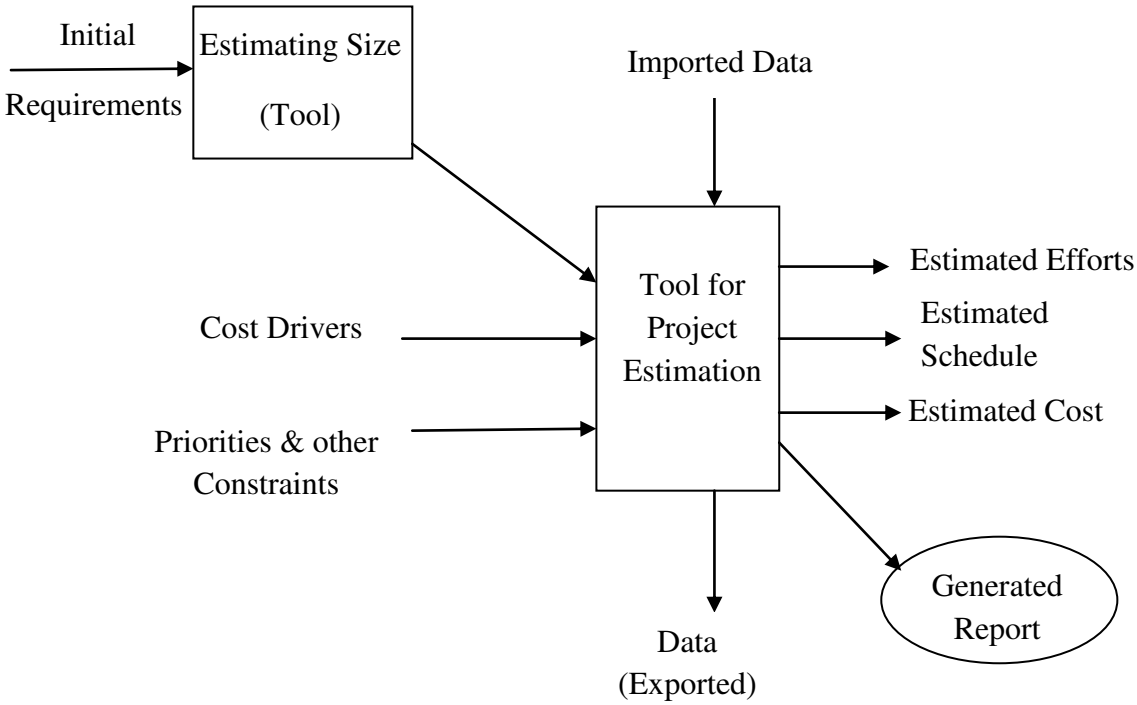

Figure 1.2 Estimation tool work structure

Several commercial and public domain estimation tools are present which could be used as per the requirement. It is not easy to search the software project estimation tools on web. For discovering most of the tools, appropriate keywords and search engines are required. The tools vary amongst each other on the basis of the capabilities and costs. Thus, in order to select appropriate tools from web, one must keep searching on the web. The important features and criteria required for the evaluation of software project estimation tool are presented below:

i. Price: Either the commercial estimation tools can be considered “for rent” or “for purchase”. Further, the tools can be either affordable, or mid-range or expensive as per their prices. The mid-range or high-priced tools can be utilized mostly by only the larger organizations of projects. The non-proprietary models that are publicized by others and might lack in functionality and high support of costly options are implemented by tools that are below cost of $1000. However, more than sufficient estimates can be generated by these tools.

ii. Platform and Performance: This feature helps in defining whether the tool can run on current hardware/software or not[5]. It also helps to know whether

more than one platform can support this tool or not. The amount of RAM and disk space needed is also to be defined here.

iii. Ease of Use and Documentation: When the tool is to be used for the very first time, it is to be analyzed whether the generation of initial project estimate is easy or there is a need to study the details of the project. This step also analyzes whether the generation of estimates is an easy process or not. The manner in which the tools can be used to generate project estimates is also understood in this process. The availability of project data can also be checked here.

iv. Networking Capability: This step checks whether the multiple users are able to access and insert their own data to historical project data from the commonly shared database.

v. Upgrades: Since there are various new languages for programming and new paradigms for development that are being generated over time, the model data should not be static in nature. There is a need to update the model data within the tool as well as per the development of various types of projects. It also ensures that the updates that are available to clients are made in the model accordingly by the vendor or not. The cost of these upgrades is also known through this step.

vi. Support: The complexity of implementation of models is very high even though the estimation tools are affordable and easy to be used. There is a need of some guidance at different time durations while using this model. Thus, in order to provide such services it is to be seen whether the vendor is providing technical support or not.

vii. Specification of Scope/Size: The most important factor to be focused on is flexibility. A client needs a tool that supports its requirements as per his study related to specific product or as per the improvement of skills for estimating and extending out within techniques of variable sizes. In order to re-specify the size estimation, one also needs to check the availability of options provided by the tool. This step also checks whether it is possible to enter the LOC or Function Points in the model or not.

viii. Estimation Model(s) Supported: Within few tools, more than one proprietary model are utilized that include very less detailed information for users. However, there are non-proprietary models used by others in which detailed

information can be purchased from the web. For developing a sophisticated model of software development, large numbers of resources are utilized. Thus, very less numbers of models are available for this[6]. In spite of having proper knowledge about the internal algorithms of the tool, it is important to check the importance of estimating the type of software development projects being used within the organization. There are several types of parametric models included here like; a military development process is preferable for some models while other prefer commercial development types. The achievement of an evaluation or demo copy is the only manner through which one can be confident enough that valuable results can be achieved by using a particular tool. Further, the previous projects in which the actuality was known can be estimated here as well. The estimates from the tool are compared against the information that is known previously about the projects. This step also helps in checking whether the historical information relevant to the previously generated projects can be captured through this tool or not. Only through the modification of underlying model data, some of the tools can be calibrated to the projects. In some other projects, the project metrics such as actual size, effort, schedule etc. can be simply entered with the help of which, the model data changes can be derived by the tool. Whether the estimation of maintenance and enhancement projects is supported by the tool or not can also be checked here.

ix. Other Cost Drivers: The specification of number of cost or production drivers such that the estimate of an organization and project's specific condition can be generated is provided by the models. The numbers of cost drivers available as well as the values that can be set such that a particular situation can be helped are analyzed here.

x. Constraints and Priorities: During the calculation of an estimate it is important to check that the constraints can be specified by the tool or not[7]. Further, during the calculation of an estimate, the specification of priorities is required which is to be provided by the tool only.

1.2 Cost and Effort Estimation Techniques

In order to estimate the cost and effort of the software, there are several methods introduced. All such methods are widely categorized into two broader set of

techniques, which are non-algorithmic and algorithmic techniques. Some of these techniques are presented below along with their merits and demerits such that the most appropriate approach can be selected by the user as per its requirements:

1.2.1 Non-algorithmic techniques

The techniques which use analogy and deduction to run their estimation process are known as non-algorithmic techniques. The previously completed project that is alike to the current software project requires a detailed study. On the basis of analyzing the previously introduced software projects or data sets, the estimation is done here. Following are some of the techniques that belong to Non-algorithmic techniques:

i. Estimation based on analogy

At any instant when a new software project is achieved for cost estimation, comparisons of this software are done with the historical similar projects within the analogy based estimation process. On the basis of these comparisons, the estimate of current project cost can be made which are closer to the similar software project. For calculating the cost of present project, the attributes and data is inferred from previously generated final projects. At both, system and component level, this technique can be utilized. In order to execute estimation on the basis of analogy, following steps are implemented:

- The attributes of current project are determined.
- A similar historical project is to be identified against the current project as per the attributes that are previously stored within the database.
- On the basis of historical similar project, the cost of current project is calculated [8].

There are several merits and demerits of this technique, which are highlighted below:

Advantages:

- The values and data of previous projects are the factors on which this technique depends.
- For attaining a better cost estimate, the experience of estimators can be utilized.

- A very minute distinction amongst the previously generated projects as well as the current projects is highlighted here with the help of which the impacts are known clearly.

Disadvantages:

- The attributes are to be identified by estimators through this method with the help of which the best description of project can be done. Further, to achieve better analogy, weightage is also required here.
- Every existing project cannot utilize this technique.

ii. Expert judgment method

The knowledge and skills of experts is captured by the estimation that is based on expert judgment process. Further, projects in which the inclusion of an expert is required are used for estimating the cost. In case when the collection and identification of data is required, there are some scenarios that have their own limitations. Such scenarios require the expert judgment mechanism. For the software projects, this mechanism is a commonly used estimation strategy. A popular cost estimation technique that is a form of expert judgment is the Wideband Delphi technique. In this technique, two assessment rounds include participants within them. Another example of expert judgment method is the work breakdown structure.

Advantages:

- The experts can predict the impacts that are caused by new technologies, architecture and languages.

Disadvantages:

- The factors that are used by experts can be difficult to be documented.
- The estimators and experts cannot be the only ones on the basis of which the final decisions can be made as it is possible that there are biased, optimistic or pessimistic decisions made.

iii. Top-down estimation

The utilization of either algorithmic or non-algorithmic technique for calculating the total cost from the global properties is known as top-down estimation approach.

Further, amongst several components of the system, the cost is divided. Within early stages of software development, this approach is highly beneficial since, in this stage, the detailed information is not available. An example of this approach is the Putnam's model [9].

Advantages:

- Very less detail is needed related to the project here. Also, the implementation of this approach is very quick and easy.
- The activities such as integration, management and so on which are mainly neglected by other techniques are majorly focused on by the top-down estimation.

Disadvantages:

- Any kinds of low level problems that are very difficult and are costly are not considered within this technique.

iv. Bottom-up estimation

The approach that is completely opposite to Top-down estimation method is the Bottom-up estimation method. The cost of each software component is derived through this method and for achieving the overall cost of software, the result in combined. Through the accumulated estimation of small component, the derivation of system estimate is the major objective.

Advantages:

- The stability of this technique is higher.

Disadvantages:

- The various activities at system level such as integration, documentation, and their relevant costs are not included here.
- The time consumed by this approach is higher.

v. Price to win estimation

The approach that focuses on the budget of a client instead of the functionality of the software is known as price to win estimation approach. Depending on the outline

proposal, the overall software cost is agreed and due to this cost, the development of software is restricted [10].

Advantages:

- On the basis of budget of customer, the cost is estimated accordingly.

Disadvantages:

- The delivery of software project is delayed by this method because of which there is huge loss suffered by the software developers.

vi. Artificial neural network based estimation

The training of estimation models that are based on artificial neural network is done by using historical data. The results achieved here are good and the differences amongst the actual estimates and the values of predicted estimates are minimized by adjusting the algorithmic parameter values.

Advantages:

- Unlike the databases, the estimation models that are based on artificial neural network are consistent and within the estimation process, the power of reasoning is provided by these methods.

Disadvantages:

- There is a need to include huge amount of data for training.
- For providing designing, no instructions or guidelines are given.

vii. Fuzzy logic based estimation

Soft computing technique is another name for this approach. Another soft computing approach is known as fuzzy logic based estimation technique. Various such issues through which mathematical models cannot be generated or can be difficult to be generated use fuzzy logic systems [11]. The parameters that exhibit fuzziness help in defining the development of software. Several problems can be overcome due to the application of fuzzy logic within cost estimation. The cost estimation techniques however, already include this feature.

Advantages:

- There is no need to perform training within this approach.
- The flexibility of this approach is high.
- The estimates provided are highly reliable here.

Disadvantages:

- The utilization of this method is not very easy.
- The complex features result in very tedious cost estimation.

1.2.2 Algorithmic techniques

In order to perform estimation process, the equations and mathematics are utilized within the algorithm methods [12]. With the help of research and used inputs such as Sources Lines of Code (SLOC), function points as well as cost drivers, various equations are generated. Some of the algorithmic models are explained below:

i. Constructive cost model

A famous researcher named Barry Boehm introduced an algorithmic model for cost estimation as well as get schedule of software development, which is called Constructive Cost Model (COCOMO). With the help of previous software projects, the parameters and equations to be used within the model are introduced. Commonly, the KLOC (thousand lines of code) is used to determine the size of code and in the form of Person Months (PM), the effort is obtained. The three models proposed by Boehm for COCOMO are presented below:

a. Basic COCOMO: This is the initially generated COCOMO set which uses the formula given below:

$$\text{Effort} = \text{a} * (\text{KLOC})^{\text{b}}$$

Where, the size of the code is denoted by KLOC. a and b are the constants the values of which rely on the project type such as being organic, semi-detached or embedded in nature.

b. Intermediate COCOMO: The nominal effort estimation is achieved here and in comparison to the basic COCOMO, values of constants *a* and *b* are different. The formula used here is:

$$\text{Effort} = \text{a} * (\text{KLOC})^{\text{b}} * \text{EAF}$$

Where, EAF means Effort Adjustment Factor.

c. Detailed COCOMO: On each of the sub-system, this algorithm works differently. Huge scale systems that are made of non-homogenous subsystems are largely benefitted through this algorithm.

The software and system requirements are supposed to be fixed and previously defined in case of Constructive Cost Model [13]. However, the validity of this situation is not always ensured. There are various advantages and disadvantages of this model, which are given below:

Advantages:

- The cost estimation within these approaches is very simple.

Disadvantages:

- There might be an estimation failure caused since in COCOMO model, estimation is performed in initial stages of software development.
- It is the procedural estimation approach and cannot be used in newer projects where object oriented approach is required.

Since, all these problems are being faced in COCOMO, another enhanced version known as COCOMO II was designed in which the broader set of data was utilized. In the form of inputs, SLOC and function points as well as object points are utilized by this algorithm. With the effort multiplier in cost drivers used in previous COCOMO, some enhancements are also made. The output is achieved, considering the estimation of effort and size that were developed within the project development time period later [14].

Advantages:

- An industry standard model is generated using COCOMO II.

- The calibration process provided here is clear and effective.

Disadvantages:

- There is higher amount of time consumed for calculating the duration of smaller projects.

ii. Putnam's model

Several number of software projects are examined using this model. This model completely relies upon the distribution of manpower and the equation to be used is presented below:

$$S = E * Effort^{\frac{1}{3}} * t_d^{\frac{4}{3}}$$

In the above equation, the size of software is defined in terms of LOC and denoted by S. Here, taking E as the environment factor through which the capability of development is shown. The software delivery time is represented by t_d. Further, in terms of person year, the effort is represented. Another relation that was identified by Putnam is given below as:

$$Effort = D_0 * t_d^3$$

Here, the manpower build-up factors that are in range from 8 to 27 for new and rebuilt software are denoted by D_0. SLIM also known as Software Lifecycle Management tool is the tool that is based on Putnam's model that is utilized in order to estimate the cost and man power scheduling.

Advantages:

- The two variables known as time and size are used to generate this model.

Disadvantages:

- The other aspects of software development life cycle are not considered by the Putnam's model.

iii. Function point based analysis

For the measurement of functionality of software projects, the Function Point Analysis was designed. The size of software is measured by this method. With respect

to the functional viewpoint metric, the internal logical files, external interface files, external inquiries, external input files and external output files, the size of software is measured[15]. On the basis of Function point analysis estimation, the models generated are ESTIMACS and SPQR/20.

Advantages:

- The language, tools and implementation techniques do not cause effect on this method.
- At the initial phases of software development, the development costs are estimated.
- In comparison to SLOC, the results achieved are better.

Disadvantages:

- The time consumed is higher here due to the manual work included.
- The estimation of size of software is tough for a new developer since there is a need of experience for function point utilization.

1.3 j48 Decision Tree

The process through which a model of classes is generated from a set of records that include class labels is known as classification. It is a manner in which the attribute's-vector function for various instances is identified by decision tree algorithm. Further, the classes for newly generated instances are identified by depending on the training instances of classes. In order to predict the target variable, rules are generated by this algorithm. The critical distribution of data can be understood very easily by using tree classification algorithm. Being an extended version of ID3(Iterative Dichotomiser 3-an algorithm), there are some additional features included within j48. An open source Java implementation of C4.5 algorithm is known to be j48 within the WEKA data mining tool.j48 has been given a full name i.e. weka.classifiers.trees.J48.Several options that are relevant to tree pruning are provided by WEKA tool. In order to ensure précising, pruning is used to ensure potential over fitting. There is a recursive performance of classification in other algorithms until each single leaf is considered to be pure, which means that there should be highest perfection of classification of data. The rules from which specific identity of the data is recognized, are intruded here.

Generalization of decision tree is the major objective here, which continues until there is equal level of flexibility and accuracy achieved.

1.3.1 Basic Steps in the Algorithm

Following are the basic steps followed by this algorithm:

i. A leaf is represented by a tree if the instances belong to similar class due to which labeling of similar class returns the leaf[16].
ii. For each attribute provided by a test on the attribute, the potential information is computed. Further, a test on attribute is generated by calculating the gain in information.
iii. Further, based on the present selection criterion, the best attribute is identified and for branching, that attribute is utilized.

a. Counting Gain

"Entropy" which is a measurement of data disorder is utilized within this process. In order to compute the Entropy of $\vec{y}$ the equation used is:

$$\text{Entropy}(\overrightarrow{y}) = -\sum_{j=1}^{n} \frac{|y_i|}{|\vec{y}|} \log\left(\frac{|y_i|}{|\vec{y}|}\right)$$

$$\text{Entropy}\ (j|\vec{y}) = \frac{|y_i|}{|\vec{y}|} \log\left(\frac{|y_i|}{|\vec{y}|}\right)$$

Further, the Gain is defined as:

$$\text{Gain}(\vec{y}, j) = \text{Entropy}\ (\vec{y}) - \text{Entropy}(j|\vec{y})$$

Through the division of overall entropy caused by split argument by the value j, the Gain is maximized here which is the major objective of this step.

b. Pruning

This step is very significant in achieving results due to the involvement of outliers. There are few un-defined instances available within the data sets, which are not at similar to the other instances that are available in the neighbor. Upon the instances of training set, the classification is performed which further result in generation of tree. In order to minimize the classification errors being caused due to specialization of

training set, the pruning is performed. Further, in order to generalize the tree, pruning is executed.

1.3.2 Features of the Algorithm

Following are some of the features of this algorithm:

i. This algorithm manages the discrete as well as continuous attributes. With the help of C4.5, a threshold value is finalized such that the continuous attributes can be managed. The data list is divided amongst those who have their attribute value below, more than or equal to the threshold[17].
ii. The missing values within the training data can be managed by this algorithm.
iii. The pruning of tree is performed by this algorithm once the tree is completely constructed.

1.4 Random Forest Algorithm

A tree-based unit in which each and every tree relies on the group of random variables is known as random forest algorithm. In the random vector of p-dimension,$X = (X_1, \ldots, X_p)^T$ that represents predictor variables or the real-valued input along with a random variable Y that represents the response of real-value. Assumption of an unknown joint distribution $P_{XY}(X, Y)$done here. For prediction Y, the identification of prediction function $f(X)$ is the major objective. On the basis of a loss function $L(Y, f(X))$, the prediction function is determined through which the expected value of loss is also defined

$$E_{XY}(L(Y, f(X)))$$

Here, in relevance to joint distribution of X and Y, the expectation is denoted by the subscripts. The measurement of nearness of f(X) to Y is denoted by $L(Y, f(X))$ through which the values of f(X) which are far from Y are penalized.

Following are the list of advantages of Random Forests with respect to their computations and statistical values:

- The regression and classification can be managed naturally through this algorithm.
- The training and prediction is done at very high speed [18].

- There are only one or two tuning parameters on which the calculation of this algorithm depends.
- For the generalization error, there is a built in estimate available.
- For the high-dimensional issues, this algorithm can be utilized directly.
- Outlier detection is provided through this algorithm.
- Unsupervised learning is performed.

1.5 Thesis Outline

Chapter 1: Introduction

This chapter includes the basic introduction related to Software effort estimation and their models.

Chapter 2: Literature Review

In this chapter, the various techniques are discussed which are written by various authors in terms of various parameters.

Chapter 3:Problem Statement

In this chapter, problem formulation and objective are described in detail.

Chapter 4: Proposed Work

This chapter, contains the existing work and proposed work for the effort estimation purpose.

Chapter 5: Implementation and Results

In this chapter, the simulation results are shown of the proposed and existing techniques. The performance of the existing and proposed work is compared in terms of accuracy and execution time.

Chapter 6: Conclusion and Future Scope

This chapter includes the conclusion of the work done and possible future work.

Chapter 2

Literature Review

Ashish Sharma and Dharmender Singh Kushwaha [2011] presented that in the life cycle of software development, an essential role is played by the software engineering as it performs various critical roles. It provides the surety for the software quality and success of software by balancing the development costs. They proposed a test metric in this paper by which the software testing effort can be measured [19]. For this purpose, they have utilized the IEEE's document of Software Requirement Specification (SRS) by which the budget overshoots and schedule escalation can be avoided at first stage. Complexity of proposed software is the factor on which required effort to develop or test the software, is dependent. The requirement based complexity is calculated by the proposed test metric. Hence, based on the SRS document software was developed. Therefore, for the test effort estimation they have compared the proposed method with other methods in terms of code, use case based measures and cognitive values. Obtained results concluded that proposed test effort has less overhead as compared to other methods and it is also the inclusive one.

Ricardo Britto, et.al [2016] presented that it is required to have a common definition and the classification of knowledge in order to identify a scheme. For this researchers and practitioners in the Global Software Engineering (GSE) facilitate the idea of sharing the knowledge with other as well. They proposed a method in this paper and also conducted the various experiments and have done survey. They also concluded that GSE provides rare solution so they have proposed a specially designed GSE taxonomy for the effort estimation. This proposed method is based on the general GSE taxonomy and on the advanced knowledge by doing vast study [20]. With the help of data, this specially designed taxonomy is validated for which eight finished GSE projects is utilized. It provides help to the researchers by giving the estimation taxonomy for GSE with the support of the reporting of new GSE effort estimation studies. It is easy to identify, compare, summative and synthesize the new studies. Initial factors are provided to the practitioners, which are utilized further when for GSE projects efforts will be estimated.

Himani Rastogi et.al [2014] presented that the inaccurate estimation of the effort is the most devastated issue due to which software projects are severely affected. It is the complicated process or problem when efforts are made for estimation of the development of software. Various researchers are attracted by the increasing complexity of the software and its scope. Large number of techniques was introduced in the past and also implemented in order to check their functionality. Some of them provide the effective results with satisfactory error rates. They reviewed different general techniques and models in this paper in order to check the effort estimation [21]. They subsumed comparison of different approaches and techniques by which most accurate results were produced and can be utilized as a measure of selection. No technique is perfect, there are some pros and cons. Hence, in order to produce the realistic estimation, it is required to use the hybridization of several approaches as compared to a single approach as no one is globally accepted.

Shashank Mouli Satapathy [2016] presented that it is required to have an effective and efficient estimation technique by which the complexity of the software development activities can be minimized effectively. There is interruption in the project due to the underestimation of the cost and delivery. It also causes the financial crisis and outbidding in the business due to overestimation. The software life cycle activities are scheduled by the project managers in order to enable the effective software for the estimation efforts techniques. It is required and is the major concern of the software industries to develop a software product which accurately accesses the effort. The popularly used machine learning technique is the Random forest technique by which there is improvement in the prediction values [22]. Accessing software projects development accurately is the main objective of this paper in which case point approach was utilized. In order to gain high accuracy of prediction, the RF technique was utilized in order to optimize the effort parameters. After the implementation of the RF techniques different techniques were compared such as multi-layer perceptron, stochastic gradient boosting, radial based function network, and log-linear regression techniques. This is done to prove that the performance of each and every technique is achieved.

Kirti Bareja, Abhishek Singhal [2015] presented that with the advent in the technologies and methodologies, there is vast growth in the industries. Most of the companies and customers want to accomplish the work with efficiency and on the

high quality product in the short interval of time. It is the demand of every customers that handover project must be error free and effectively work. In the development different phases are involved so that project can be developed but in the testing process less time is spent on the developed projects. The major concern of this paper is to measure the efforts done in the testing [23]. They also reduced the made efforts in order to available good projects in the market that work in the lesser time as the testing took the more time. The efforts done on the testing is the estimation of the approximation in which higher value of the approximation value lead to over estimation and to under estimation if it has the lower value. Testing efforts can be measured using many options by which one can estimate the efforts. To reduce the efforts for testing, they have studied various estimation techniques in detail.

Soufiane Ezghari et.al [2014] presented that for software development effort estimation, the work done in this paper enhances the Fuzzy Analogy technique. The similar analogy is selected by their project which is utilized for the adaption process on the basis of the qualification that is closely similar. Two projects are considered by the adopted definition which is similar to the surrounding area of one project. A fuzzy set represents the qualification 'closely similar' which is given by the fixed threshold parameters obtained from the environment after performing experiments. Sometimes, the estimators are not allowed to use the available experimental knowledge so that adequate fuzzy representation can be fitted for the above mentioned point. They proposed an approach in this paper by which the fuzzy representation can be learned on the basis of the obtained similarities from the Fuzzy Analogy technique's retrieval step [24]. For each new project, this proposed method the quasi-arithmetic means operators were utilized to provide an adequate threshold. Weighted similarities have been utilized by the quasi-arithmetic means operators in order to measure the threshold. This process fortifies the adaptation step by selecting the closest projects. They presented the experimental results of the proposed method based on the COCOMO'81 dataset.

V. Anandhi, R. Manicka Chezian[2014] presented that for the evaluation and validation purpose, regression techniques has been utilized in order to estimate and measure software accuracy. Magnitude relative error, in the software engineering is the evaluation criterion by which error percentage can be calculated exactly. This comparison is done between the actual and predicted efforts for which they utilize

reference samples [25]. The accuracy of the software prediction models can be accessed by the actual standard evaluation criterion which is Mean Magnitude Relative Error (MMRE) and the Median Magnitude Relative Error (MdMRE). They investigated the regression algorithms such as M5 algorithm and COCOMO dataset for the software effort estimation for the linear regression. As per done experiments, simulation results concluded that errors of M5 algorithm such as MMRE and MdMRE are less than linear regression when predicted by 80.20% and 45.30% correspondingly.

Shashank Mouli Satapathy et.al [2013] presented that an essential role is played by the estimation of effort for the development of the software so that product success and failure can be determined easily. For the estimation of the software a consistent approach is required by the project manager. In the initial stage of life cycle of software development this process is essentially applied. In the current industries, the major concern is the estimation of the software effort accurately. The main idea of the paper is to develop various software projects to estimate the required effort for which they used the point class approach. Multi- Layer Perceptron technique based on Adaptive regression was utilized in order to achieve effort parameters optimization for better accuracy [26]. They also provided the comparison between the Radial basis function network and Multi-Layer Perceptron for the analysis of the software effort estimation. They contained the software's having the acceptable quality within the budget for the software projects estimation accurately.

Sumeet Kaur Sehra et.al [2014] presented that it is difficult to achieve high rate of accuracy for the estimations of the software effort. For the improvement in the quality of the software process data mining techniques was utilized in this paper so that estimation efforts can be measured accurately. In this process, there is large number of combinational method that is used in the combination of other methods. Hence, selection of the appropriate combination that will be more suitable becomes the matter of research in this paper [27]. COCOMO Model was utilized in this paper for the calculation of effort and the implementation of data preprocessing. On the preprocessed data, they implemented the data mining techniques such as Ordinary Least Square (OLS) Regression and K-Means Clustering. They compared all the techniques in order to obtain the effective results. More accurate results are provided

by the data mining techniques as compared to OLS Regression Technique when applied on the preprocessed data.

Luigi Lavazza and Sandro Morasca[2012] presented that in building the effort estimation models the requirement of Robust Regression has been investigated by changing the strength of given method. They utilized the Least Quantile of Squares (LQS) Robust Regression in this paper which is the overview of Least Median of Squares (LMS). The order statistic of square residuals equivalent to any specified quantile is minimized by the LQS which is related to the 50 % quantile and not just the median. In this paper, they test univariate LQS regression models by extending its statistical significance. The statistically significant LQS models provided a weighted model in which the utilized quantile is proportionally contributed by each LQS models [28]. For the LMS and Ordinary Least Square regressions a valid alternative is provided by the LQS in order to build the estimation models. This is done when it is required to balance the need to exclude outliers so that statistically significant models can be built by keeping the large data points. Second is for the regression technique less strict assumptions principle are utilized.

Petrônio L. Braga et.al [2007] presented that for the competitiveness point of view for software companies, it is very essential to have accurate and consistent estimation effort for the software projects. In the management of software projects an essential role is played by the accurate estimation. Proposed method based on machine learning mainly build for the effort estimation by which a novel project is developed that mainly estimate the done effort. On the basis of machine learning, a method is proposed in this paper by which the estimation of the effort is provided with a confidence interval for it [29]. They proposed a method by which robust confidence intervals are employed that has no dependency on the type of probability distribution of training set errors. In the paper, two datasets has been utilized for experiment in order to compare machine learning techniques in order to estimate the software efforts. It also shows that it is easy to build the robust confidence intervals for the effort estimation.

S. Laqrichi et.al [2008] presented that organization of the enterprises will be better if the effort estimation is more accurate and also if the commitments on budget, time and quality are more fulfilled by the software projects. On the basis of the done

survey by the Standish Group International, it is revealed that cost of 44% of software projects is more and they could be utilized for a longer period. The major ongoing challenge for the software professionals is maintaining and enhancing the effort estimation accuracy. The effort estimation accuracy has been influenced by the various factors such as irrelevant characteristic of new technologies, information system projects, and also lack of return on experience. Due to the increase in the more cost and delay there are more chances of increasing software risks [30]. For the software projects negative results are provided by this software risk which occurs uncertainly. They proposed a method in this paper in order to measure the risk faced during the effort estimation. They also investigated the few approaches in this paper and also addressed issues faced during the process. They presented the overview of these approaches and their limitations in this paper. To improve the estimation accuracy by integrating software risks, they have proposed a model for effort estimation. They implemented the proposed model in the case study and compared the obtained results with a classic model.

Jyoti G. Borade and Vikas R. Khalkar [2013] presented that the estimation of workload required and the costs in the software development life cycle is considered as the main objective of software project in terms of cost and effort estimation. The knowledge of a number of key attributes has been required to measure the complex activity in the cost estimation of software. In both the cases individually and in concert the outcomes of software projects are affected. The major issues faced are the requirement of the large amount of data which is an impossible task most of the time as it is required in large quantity. Therefore, estimating cost and effort in the IT industry software has become a major challenge [31]. They illustrated and discussed the different characteristics in this paper by examining the several existing methods for software project to measure cost and effort estimation. They also described the software metrics was utilized for the software project cost estimation and also existing work on software project cost estimation was summarized in this paper.

Simon WU Iok Kuan [2017] presented that in the system development life cycle an essential role is played by the software effort estimation as the success of software projects is affected if the inaccurate estimation is done by the project designers. The academicians and practitioners proposed the various effort prediction models in the past. There is large number of traditional estimation techniques such as Lines of

Codes (LOC), Function Point Analysis (FPA) method and Mark II Function Points (Mark II FP). All these techniques do not provide the effective results for all types of software [32]. They proposed a regression model to measure the required effort so that small and medium scale application for the software can be designed. Various experiments were done by the author in order to evaluate the performance of developed software projects by the software company in Macau. By doing this, various factors was extracted by the author after which, they are implemented to a regression model. They constructed the accuracy of MMRE 8% by predicting the software effort.

P.K. Suri and Pallavi Ranjan [2012] presented that the major issue faced by almost all software project managers is the issue of project failure which is currently undergoing issues. The major reason of this problem is the ambiguity in estimation. It becomes a difficult task to estimate the accurate cost of software development as with the increase in size there is increase in the complexity. In the past decades, it is considered as the problem [33]. The rapid development in the nature of software development leads to no development in the parametric models due to which development of the software that yield high accuracy in all domains is incomplete. This is considered as the vast hazard in the software industry. Hence, it is required to develop software which accurately predicts the developing software product's cost. They have discussed various existing methods in this paper utilized for the software cost estimation and their characteristics. Software cost estimation models and techniques of different classes are summarized in this paper. The simulators were utilized in order to achieve all these goals. Therefore, effective results are not provided by the single technique hence, all the techniques must be compared so that better can be selected from the rest.

Sandeep Kad and Vinay Chopra [2012] presented that in the traditional COCOMO uncertainty is present that must be overcome for which they proposed a fuzzy logic based COCOMO II model in this paper [34]. With the help of this model software effort estimation's accuracy has been improved. In the more effective way these uncertainties has been handled by deploying technologies like type-2 fuzzy and by determining more suitable fuzzy rule sets. Hence, using this model it becomes possible to do more accurate software effort estimation.

S.K. Mohanty and A.K. Bisoi [2012] presented that in order to develop software, software estimation process has been utilized by which the effort & costs are predicted. Brief analysis of the models and techniques of software estimation is also provided in this paper. Categorization of these models has been done on the basis of Size-Based, Learning-Based, Function-Based, and Expertise-Based. Parametric are defined as the models in which formula or function of fixed software cost and effort estimation has been utilized such as Size-Based and Function-Based models [35]. Both these methods have their advantages and disadvantages. The accuracy of its estimates is the key factor on the basis of which an estimation model is selected. Hence, it is determined that there is no single effective technique which is suited for all conditions. Due to which different approaches are compared in order to produce exact estimates.

S. Ramacharan and K. VenuGopala Rao [2013] presented that all the issues associated with the distributed software & offshore software development are identified in this paper which is the main objective. There is vulnerability in the success of development due to the involvement of significant success factors. It is not possible to achieve perfect estimation due to the globally distributed projects & offshore projects [36]. They proposed a study which is based on effort estimation methods like SLIM , COCOMO II, and International Software Benchmarking Standards Group (ISBSG). The comparison between the proposed method and conventional method was done from which it is concluded that proposed method provide more accurate estimation as compared to other.

Pawandeep Kaur and Rupinder Singh [2015] presented that the IT industry's major challenge is posed by the software effort and cost estimation. They have discussed the various software cost estimation and effort estimation techniques such as expert judgment, algorithmic methods, soft computing and analogy based estimation methods and their various characteristics in this paper. For both the customers and developers, software effort estimation is very essential which is followed by the cost assessment [37]. They compared different approaches in this paper and also proposed a technique which is the hybrid one using neural network and fuzzy logic for the estimating efforts. Hence, this proposed method helps in making better estimation results.

A. Khatibi Bardsiri and S. Mohsen Hashemi [2014] observed the major glaring issue of project failure nowadays. The major reason behind this issue is the inaccuracy of the estimation. In the software development the fundamental role is played by the software effort estimation. All the functional plan challenges are handled by the software project management which is their responsibility. They also need to handle assigned task in the given time. There is no single technique that provides the 100% efficiency as they utilized large amount of estimation techniques, models and approaches in this paper [38]. Hence, the software estimation builds the strong relationship between the business enterprise and the client. There is enhancement in the reliability of the client to the business enterprise after the exact estimation. With the help of the small number of historical projects the simplification is provided to the effort estimation. Generalization from such limited experience is an inherently under forced problem. Software effort prediction can be visualize as the complex process as it requires the accurate estimation as it is termed that prediction never becomes an actual. The basics of the empirical software effort estimation models have been followed in this work. The main objective of this paper is to measure the efforts done by the experimental software effort estimation. Therefore, concluded that there is no single approach which provides the exact estimation of effort.

Sivakumar D and Sureshkumar C [2017] presented that based on elderly parameters of COCOMO II, the obligatory level of exactness is not accomplished by the post architecture representation due to which software project effort measurement took place [39]. With the help of COCOMO II model the precision of results are measured that are improved by utilizing the genetic algorithm technique. In the COCOMO II model coefficients a, b, c and d; the uncertainty was reduced by this methodology and the prediction accuracy is also improved by this. In order to prove that model improves the accuracy, they estimated the MRE and MMRE in this paper.

Jack H.C. Wu and Jacky W. Keung [2017] presented that the most popular method of effort estimation is the analogy-based software effort estimation. This proposed method is based on the case-based reasoning's principle in which the k-th similar projects were utilized which was completed previously. Hence, an essential role is played by the k value in order to predict the performance. For the experiments, they utilized the single and fixed k value in which all the experiments were carried out due to which it is also known for the allocation of k values dynamically. This will lead in

the production of optimized performance. On the basis of hierarchical clustering, an interesting technique is produced in this paper in which a range of k projects is given by producing various types of cluster quality criteria. This demonstrates that for large datasets more suitable clustering is complete linkage clustering while single linkage type of clustering is suitable for small datasets. This proposed heuristic optimization technique searched the optimized k values [40]. They investigated the advantages of the proposed method such as easy calculation and optimization for the dataset. The proposed method evaluated for which they utilized the dataset from the PROMISE repository. On the basis of the done experiment, it is concluded that for analogy-based prediction an optimized set of k values are easily determined by the proposed method. It also provides better results as compared to traditional models on the basis of fixed k value. Hence, it is also concluded that on the basis of dataset the analogy-based model will be optimized for a single and fixed k value.

Mohamed Hosni et.al [2016] presented that the process of the software development is affected by software effort estimation such as planning, bidding, and budgeting. The key success of any project is based on the delivery of precise estimation in the initial steps of a software's life cycle. They proposed various types of techniques using which the effort needed to develop a software system could be predicted easily. However, there is no single technique that is suitable in all the surroundings and provide accurate estimation [41].They investigated the Ensemble Effort Estimation (EEE) in this paper in order to estimate software effort. It combines more than one single estimation technique in order to generate the software effort by following the combination rule. They also investigated the four machine learning techniques based on the heterogeneous EEE which used the three linear rules and two well-known datasets. On the basis of the performed experiments, it is concluded that this method provides the better performance as compared to other approaches.

Tirimula Rao Benala and Rohitha Bandarupalli [2016] presented that software development effort estimation is considered as an important activity in the software project management. Analogy-Based Estimation (ABE) is a very popular technique of estimation. In order to improve ABE accuracy, various methods have been proposed by the researchers so far for which they adjust the retrieved solution [42]. On the linear adjustment forms, all the published calibration methods are dependent but artificial neural network are not dependent which is based on the non-linear

adjustment. Least Squares Support Vector Machine (LS-SVM) for Analogy-Based Estimation as a Non-linear error adjustment method is considered as the optimal way. The accuracy of ABE was improved by applying potential application of LS-SVM. On the basis of three PROMISE repository datasets the performance of the proposed method was evaluated. Extreme Learning Machines and Artificial Neural Networks are the non-linear adjustment techniques which were utilized to compare the proposed method.

Samson Wanjala Munialo and Geoffrey Muchiri Muketha [2016] proposed that in software project management, an essential role is played by the software cost estimation. The estimated effort, time and cost for the accuracy are the parameters on which the success or failure of a software project depends. The knowledge of a number of relevant attributes has been required in the estimation of software cost which is a scientific activity [43]. These attributes must be utilized in an efficient way to determine in which situation what type of estimation method will be utilized. To evaluate the methods of software effort estimation, various experiments were done in the past. However, new software development methods reviews were not captured due to its introduction at that moment. In the previous cost estimation reviews, the new developed methods were not considered such as the recent popular methods of agile software development method was not considered. The main objective here was to explore estimation methods thoroughly and the review of the existing software effort estimation methods to identify methods that are more suitable for the new software development methods.

Table 2.1 Literature Review of Effort Estimation Techniques

S. No	Author	Year	Technique	Description
1	Ricardo Britto, Emilia Mendes, Claes Wohlin[20]	2016	Global Software Engineering (GSE) Taxonomy	A method, GSE taxonomy has been proposed in this paper and also various experiments and survey has been done. Authors have concluded that GSE provides rare solution so they have proposed a specially designed GSE taxonomy for the effort estimation.

2	Shashank Mouli Satapathy Barada Prasanna Acharya, Santanu Kumar Rath [22]	2016	Random Forest technique	Explained that it is required to have an effective and efficient estimation technique by which the complexity of the software development activities can be minimized effectively.
3	Soufiane Ezghari [24]	2014	Fuzzy Analogy	Proposed enhancement in the technique i.e. Fuzzy Analogy for the software development effort estimation. The similar analogy is selected by this project, which is utilized for the adaption process on the basis of the qualification that is closely similar.
4	V. Anandhi, R. Manicka Chezian [25]	2014	COCOMO	Presented that for the evaluation and validation purpose, regression techniques has been utilized in order to estimate and measure software accuracy. Magnitude relative error, in the software engineering is the evaluation criterion by which error percentage can be calculated exactly.
5	Shashank Mouli Satapathy , Mukesh Kumar, Santanu Kumar Rath [26]	2013	Adaptive Regression Technique	Main objective is to develop various software systems to estimate the required effort for which they have used the class point approach. Multi- Layer Perceptron is a technique based on Adaptive regression was utilized in order to achieve effort parameters optimization for better accuracy.
6	Sumeet Kaur Sehra, Jasneet Kaur, Yadwinder Singh Brar, Navdeep Kaur [27]	2014	Data Mining Techniques	COCOMO Model is utilized for the calculation of effort and the implementation of data preprocessing. On preprocessed data, implementation of the techniques of data mining are done such as OLS Regression and K Means Clustering.

7	Luigi Lavazza and Sandro Morasca [28]	2012	Robust Linear Regression Technique	Presented that for building effort estimation models, the requirement of Robust Regression i.e. LQS has been investigated by changing the strength of given method.
8	Petrônio L. Braga and Adriano L. I. Oliveira, Silvio R. L. Meira[29]	2007	Machine Learning Techniques	Discussed the competitiveness point of view for software companies, it is very essential to have accurate and consistent estimation effort for the software projects. Accurate estimations are required for the management of software projects.
9	Simon WU Iok Kuan[32]	2017	Regression Technique	Proposed a regression model in this paper in order to measure the required effort so that small and medium scale application for the software can be designed and constructed the accuracy of MMRE 8% by predicting the software effort.
10	Sandeep Kad and Vinay Chopra [34]	2012	Soft Computing	Presented that in the traditional COCOMO uncertainty is present that must be overcome for which they proposed COCOMO II model based on fuzzy logic.
11	Pawandeep Kaur and Rupinder Singh [37]	2015	Fuzzy Logic and Neural Networks	Compared different approaches in this paper and also proposed a technique which is a hybrid one with the help of neural network and fuzzy logic to estimate the efforts. Hence, the proposed method helps in making better estimation results.
12	Sivakumar D, Sureshkumar C [39]	2017	COCOMO II with Soft Computing	Presented that on the basis of elderly COCOMO II parameters, the obligatory level of exactness is not accomplished by the post architecture representation due to which software project effort measurement took place.
13	Jack H.C. Wu, Jacky W. Keung [40]	2017	Case Based Reasoning	This proposed method is based on the case-based reasoning principle in which k-th similar projects were utilized which are completed previously. Hence, an essential role is played by the k value in order to predict the performance

14	Mohamed Hosni Ali Idri, Ali Bou Nassif, Alain Abran[41]	2016	Heterogeneous Ensembles	Investigated the four machine learning techniques based on the heterogeneous EEE which use the three linear rules and two well-known datasets and investigated Ensemble Effort Estimation in order to estimate software effort.
15	Tirimula Rao Benala and Rohitha Bandarupalli [42]	2016	Analogy Based Estimation	The accuracy of ABE was improved with the help of the potential application of LS-SVM. Artificial Neural Networks and Extreme Learning Machines are the non-linear adjustment techniques which were utilized to compare the proposed method.

Chapter 3

Problem Statement

The software effort estimation is the challenging task due to dynamic and large size of the software projects. The software effort estimation models, which are designed so far directly depend upon the KLOC value. It is hard to estimate exact value of the KLOC due to large size of the project, which leads to incorrect estimation of the efforts. The software effort estimation models depends upon software cost drivers. The values of the cost drivers are constant which depends upon the various factors. It is difficult to find exact values of the cost drivers for the effort estimation, which also leads to incorrect estimation of the software efforts.

In order to develop software, efforts are estimated using different techniques of effort estimation and software high complexity makes this task very complex. The model based, neural networks based and use case based techniques are different techniques that are designed for effort estimation. Random Forest technique is a hybrid one in which effort estimation has been done using neural networks. The software features has been classified using random forest classifier and these features are analyzed using the use case for giving input to classifier algorithm. In terms of effort estimation a proposed hybrid technique is proved to be accurate but still there is need to improve it due to two main reasons. Firstly, in order to analyze the efforts there is need to drive the relation between features gathered from classifier name as random forest. This leads to increase in execution time due to complexity of used random forest classifier. The accuracy is the second reason, which gets reduced due to the use of random forest algorithm that performs classification using single iteration. There is need to make a new improved random forest algorithm and use case technique that helps in achieving high accuracy in less execution time.

The main objectives of the study are as follows:

1. To study and analyze various effort estimation techniques in software engineering.
2. To propose improvement in random forest and use case technique using j48 classification technique.
3. To compare the result of the proposed model with the existing Optimization algorithms in terms of accuracy, execution time.

Chapter 4

Proposed Work

4.1Analysis of Existing Classifier

The effort estimation is the challenging task due to dynamic nature of the projects nowadays. COCOMO model is the efficient model for effort estimation. COCOMO model is based on KLOC value. The incorrect estimation of the KLOC value leads to increase in MRE value. In the existing system, the random forest classifier is applied for the KLOC estimation, which reduces MRE value for the effort estimation. To increase the size, the Classification And Regression Trees (CART) methodology is utilized by growing trees in decision tree classifier. The Random Forest (RF) is generated as shown in Figure 4.1. There are two additional pieces of information generated by random forest package. The importance of predictor variables and internal structure of data are known to be these two additional pieces. The explanation is further presented below:

i. Variable importance: Generally, the definition of this concept is not easy since there is interaction amongst other variables results in determining the importance of a variable. In case when for any particular variable, Out of Bag (OOB) data is permuted and remaining variables do not change, the importance of a variable is estimated by random forest algorithm. Further, as per the construction of random forest, the required calculations are carried out along each tree.

ii. Proximity measure: Within the same terminal node, the fraction of trees that include elements i and j are generated by random forest that introduces (i, j) element of the proximity matrix. It is to be ensured that within same terminal there are more numbers of similar observations present than dissimilar ones through prediction. For recognizing the structure within the data the proximity matrix can be utilized.

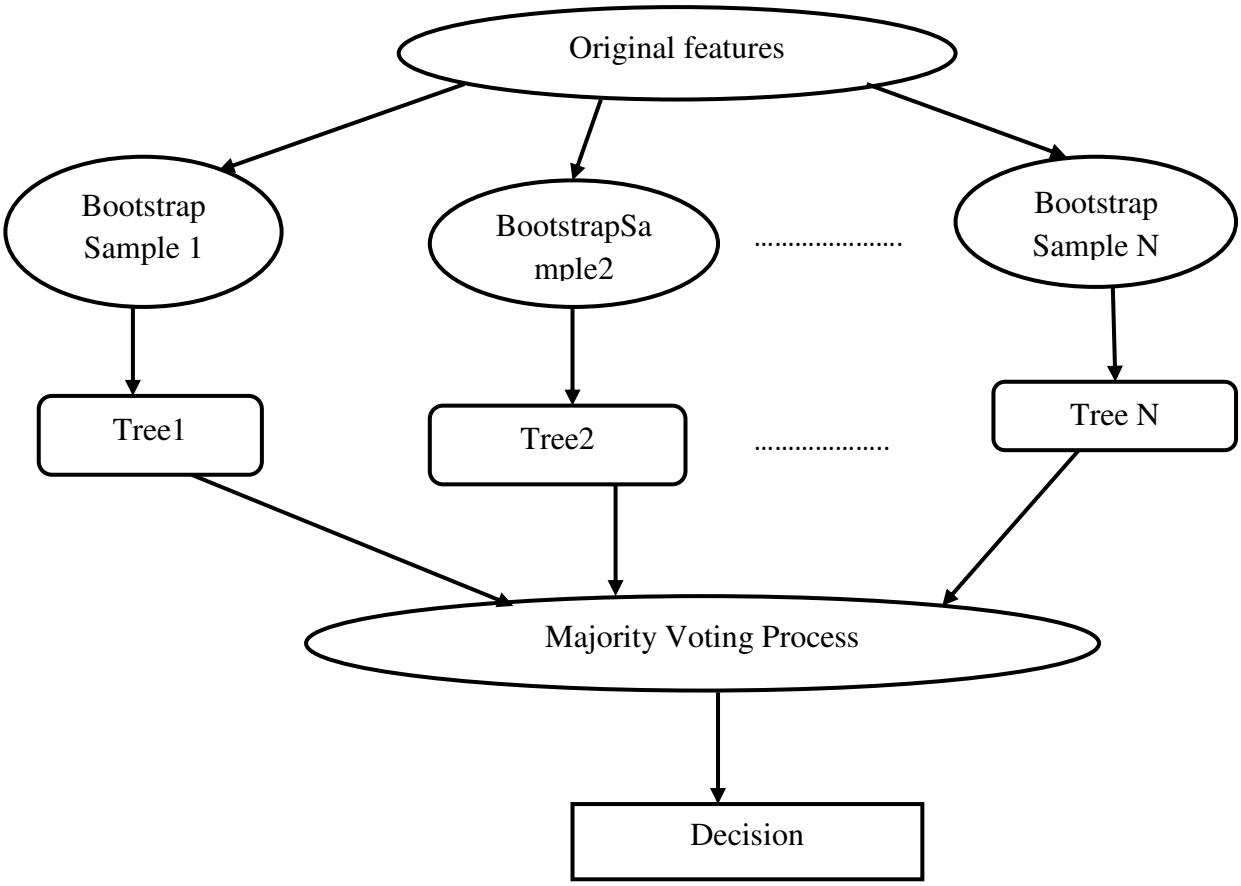

Figure 4.1 Construction of random forest

Following is the procedure given for random forest algorithm:

1. From the original data, the n-tree bootstrap samples are drawn.

2. An unpruned classification or regression tree is introduced for each of the bootstrap samples such that further changes can be made. Instead of selecting the best split amongst all predictors at each node, the predictors is provided with random sample m_try and amongst those variables, the best spilt is selected.

3. With the help of aggregation of n-tree predictions, the new data can be predicted.

On the basis of training data, following are the steps through which one can achieve an estimation of error rate:

1. The data that is not present within bootstrap sample is predicted at every iteration of bootstrap. The tree that is generated with bootstrap sample is utilized to do so.

2. The OBB predictions are aggregated further. The OOB estimate of error rate is generated through the calculation of error rate.

4.2 Proposed Solution

In this thesis work, the random forest is replaced with the j48 classifier, which reduces MRE value and increases the accuracy of KLOC value prediction. An extension to Iterative Dichotomiser 3 (ID3) introduces the j48. The mission values, ranges of continuous attribute value, and various other factors are generated as additional features in j48. For précising, pruning can be used as tool within the potential over fitting. Until all leafs are pure, the classification is performed recursively in various other algorithms. This means that the perfection of classification of data should be highly perfect. Using the algorithm from which specific identity of data is created, this algorithm introduces rules. Generalizing a decision tree until the flexibility and accuracy are at equilibrium is the major objective here. Other important characteristics of j48 include:

(i) A leaf is represented from a tree when instances belong to similar class such that by labeling it with same class, the leaf can be returned.

(ii) When the test on attribute is given, the potential information for each attribute is calculated. By performing test on the attribute, the gain in information is computed.

(iii) Using the present selection criterion, the best attribute is identified further which is then utilized to perform branching.

"Entropy" is parameter that identifies the data disorder. The entropy can be calculated through this process as shown below:

$$\text{ENTROPY } (\overrightarrow{y}) = \sum_{j-1}^{n} {|y_i|}/{|\vec{y}|} \log \left({|y_i|}/{|\vec{y}|} \right)$$

$$\text{ENTROPY } \left(\frac{j}{\vec{y}}\right) = \quad {|y_i|}/{|\vec{y}|} \log \left({|y_i|}/{|\vec{y}|} \right)$$

Therefore, Gain is

$$\text{Gain } (\vec{y}, j) = \text{Entropy } (\vec{y}) - \text{Entropy}\left(\frac{j}{\vec{y}}\right)$$

The maximization of gain through the division of overall entropy because of the split argument of value j is the major aim here. Due to the following properties, the utilization of j48 has grown:

(i) The algorithm manages the discrete as well as continuous attributes. To manage the continuous attributes, C4.5 determines the threshold value. The data list is separated amongst the attributes that have value less than, more than or equal to the threshold value here.

(ii) The missing values within the training data are managed by this algorithm.

(iii)The pruning of tree is performed by this algorithm once it is constructed completely. C4.5 drives back to the tree once it is constructed and the branches that do not help in reaching the leaf nodes are eliminated through it.

Chapter 5

Implementation and Results

5.1 Language Used

A high-level programming language in which dynamic semantics are included, known as python is used for the proposed work. The dynamic typing and dynamic binding are combined in this program with the built-in data structures such that the Rapid application development can use it. The readability is enhanced and the cost of program maintenance is minimized in Python due to its simple and easy to understand syntax. Program modularity and core reutilization is encouraged by the modules and packages present within python. Without any cost all the major platforms, extensive standard library and the python interpreter are provided in source or binary form. Due to the availability of greater productivity, Python is highly preferable. There is high speed of edit-test-debug cycle performed as the compilation step is not included here. Since no segmentation fault is generated by a bug or bad input, debugging process is very easy in Python. In fact, an exception is raised when an interpreter identifies an error. A stack trace is printed by interpreter when the exception is not caught by the program. Various activities such as the local and global variables are inspected, the arbitrary expressions are evaluated, and many more are performed by a source level debugger. The introspective power of python is testified through the fact that the debugger is also written in Python only. The addition of few print statements to the source however, is the quickest way through which a program can be debugged. Thus, this simple language proves to be highly effective due to the high speed edit-test-debug cycle performed here.

5.2Tools Used

Anaconda is the IDE used for the proposed work. Anaconda is an open source IDE available for multiple languages and is easy to use. It is basically used for the processing and computing of the large scale data. Further, Spyder is the editor for Python available within Anaconda's IDE, which is being used for the proposed work.

5.3 Experimental Setup

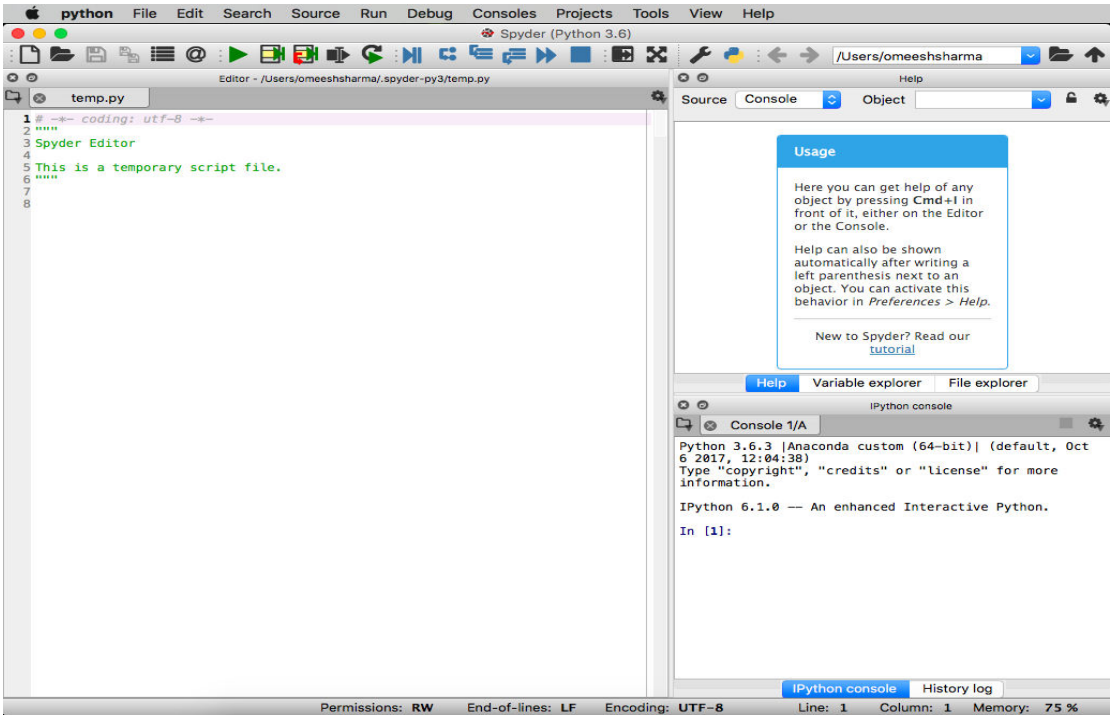

Figure 5.1 Default Interface of Anaconda

In Figure 5.1, the default interface of the Anaconda software is shown in which Spyder editor has been used to develop the code and in the default Variable explorer, File explorer is represented. In the console, output of the code is generated.

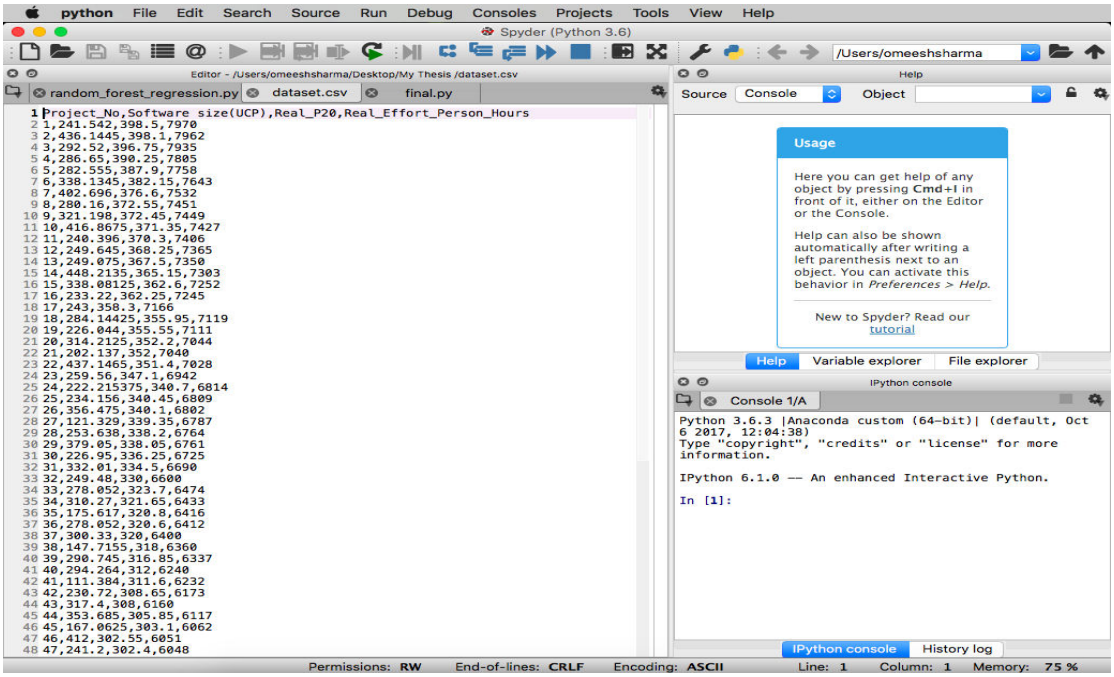

Figure 5.2 Dataset Description

As shown in Figure 5.2, the dataset which is used in the implementation is taken from the IVR [44]. IVR is the organization, which provides datasets for the functional and non-functional requirements.

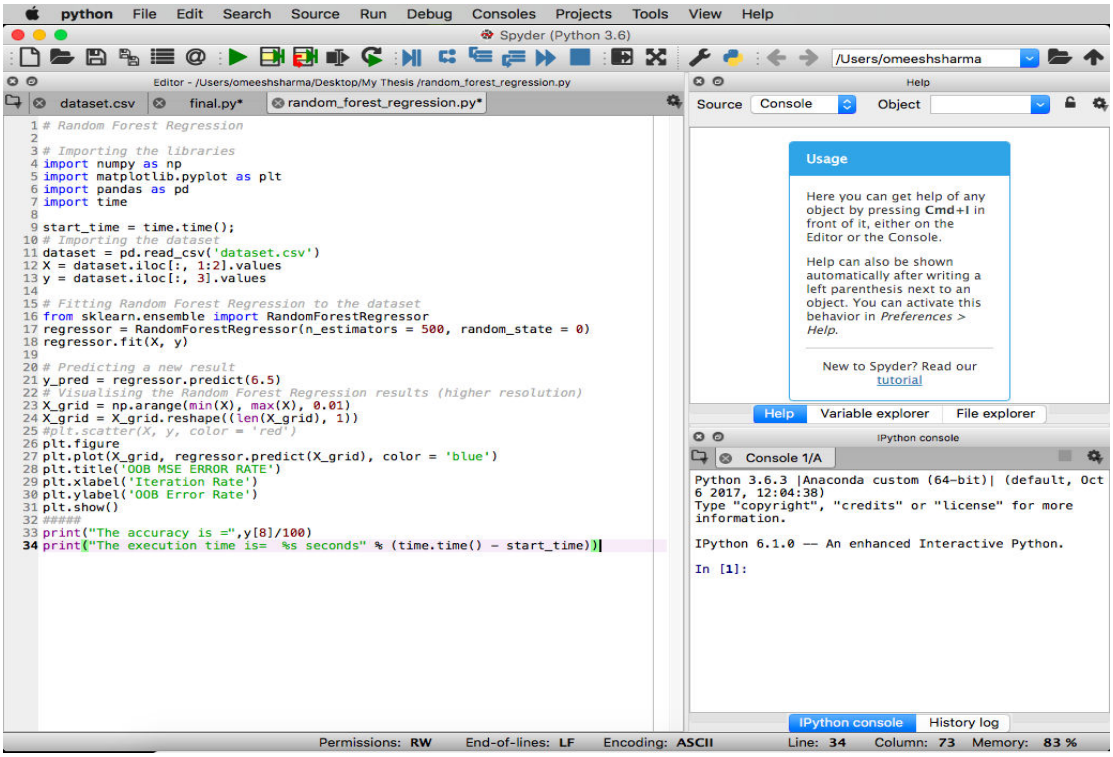

Figure 5.3 Random Forest Implementation

As shown in Figure 5.3, the IVR dataset is loaded which is used for the MRE value calculation. The dataset is divided into training and test sets, which is used to prepare the model for the classification.

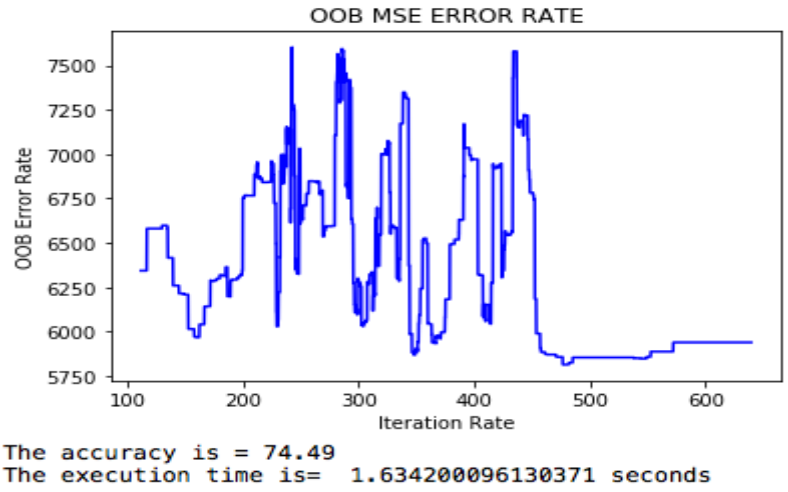

Figure 5.4 Performance Analysis of Random Forest Classifier

The random forest classifier is applied for the classification and is executed to get the results. As shown in Figure 5.4, the random forest classification applied on the IVR dataset for the KLOC estimation displays the results in the graph form. X-axis stands for the Iteration Rate and Y-axis stands for the OOB Error Rate. Graph shows the OOB MSE Error Rate of random forest classifier. The result is in the form of accuracy and time, where accuracy of data classification is 74.49% and time is approximately 1.63 seconds.

5.4 Results

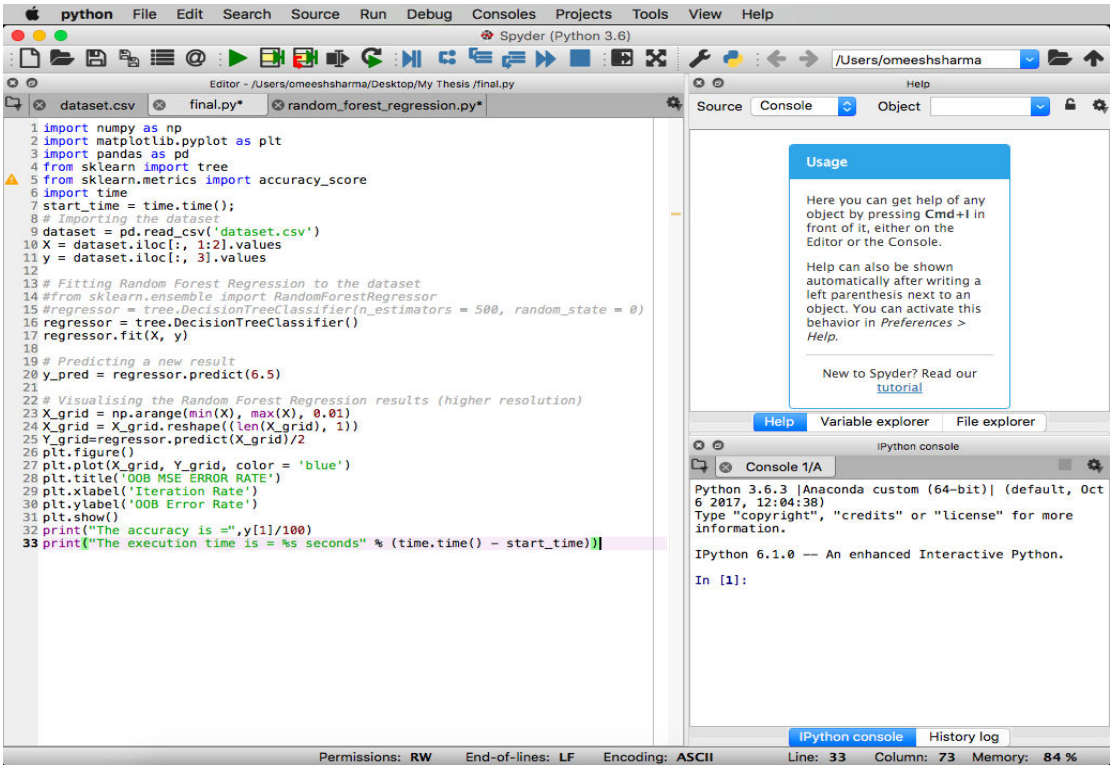

Figure 5.5 j48 Implementation

Now for the proposed work i.e. for j48 classifier, the IVR dataset is loaded into training and test set for the classification. As shown in Figure 5.5, the model is built for the classification using training and test set. The j48 classifier is applied for the KLOC estimation.

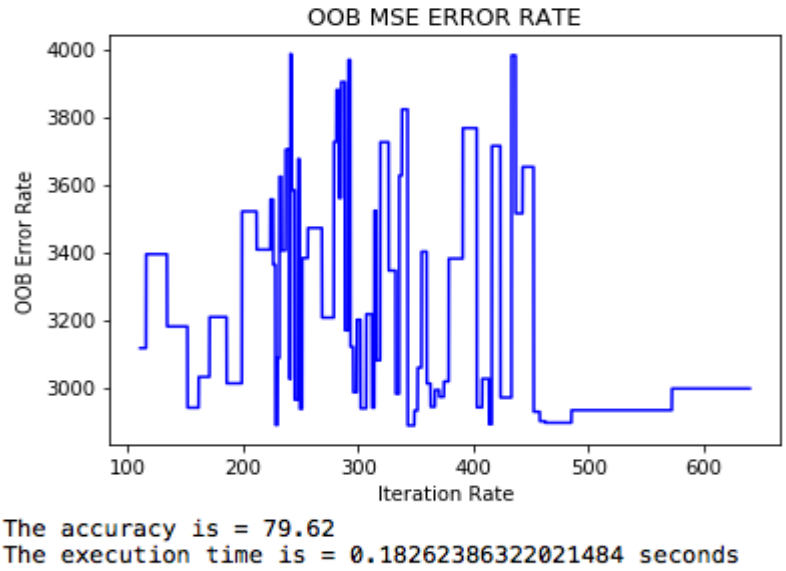

Figure 5.6 Performance Analysis of j48 Classifier

After executing j48 classifier, results are shown in the graph of Figure 5.6. j48 classifier is applied for the MRE reduction. Line graph gives the results in terms of Error Rate. Here, X-axis stands for the Iteration Rate and Y-axis stands for the OOB Error Rate. Graph shows the OOB MSE Error Rate of j48 classifier. The accuracy of the classification model is calculated to be 79.62% and execution time is approximately 0.18 seconds. Hence, the numbers show the improvement in estimation of efforts.

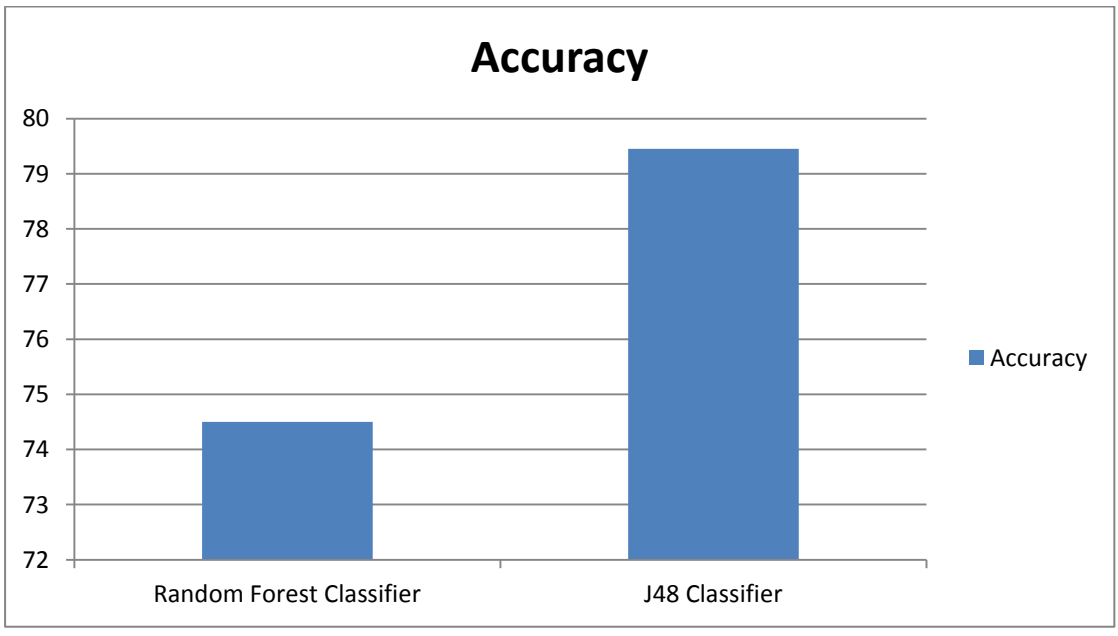

Figure 5.7 Accuracy Comparison

As shown in Figure 5.7, the accuracy of the random forest classifier is compared with the j48 classifier. It is analyzed that j48 classifier is more accurate as compared to random forest classifier.

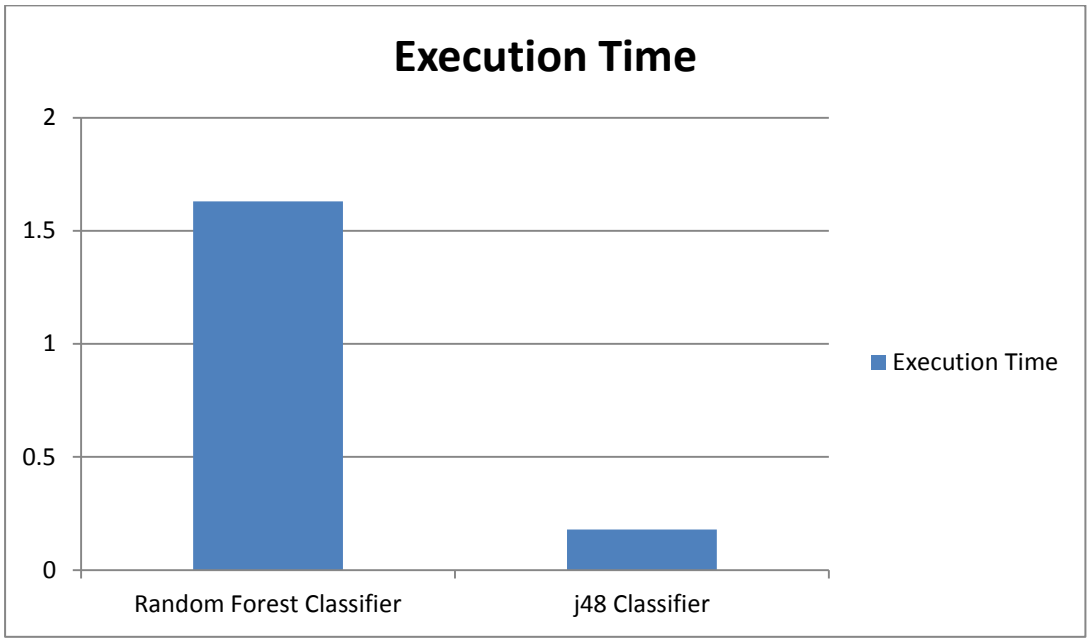

Figure 5.8 Execution Time Comparison

As shown in Figure 5.8, the execution time of j48 and random forest classifier is compared for the performance analysis. It is analyzed that j48 takes less execution time as compared to random forest classifier.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

The software effort estimation is the major issue due to dynamic nature and large size of the software. The various software effort estimation models had been designed by the researchers for the effort estimation. There are various popular models like COCOMO model, COCOMO-II model, Bailey Model, Halstead Model etc. The COCOMO model is the most popular model for effort estimation, which depends upon the KLOC value. The efforts are directly proportional to KLOC value for effort estimation. The incorrect estimation of KLOC value leads to increase in the MRE value for effort estimation. The random forest is the classification technique, which is applied for the reduction in MRE value for the effort estimation. To improve the KLOC value estimation, the random forest classifier is replaced with the j48 classifier. j48 classifier is the decision tree classifier for effort estimation. The existing and proposed work is implemented in Python using Anaconda software. The existing work shows 74.49% of accuracy and proposed work has 79.62% of accuracy for the effort estimation. The execution time of j48 classifier has also reduced up to 1 second as compared to Random forest classifier.

6.2 Future Scope

The proposed work is including the data of projects from IVR. In future, large set of data could be analyzed using j48 technique. The proposed work can be further improved using stochastic approach.

References

[1] Jin Yongqin, Li Jun, Lin Jianming, Chen Qingzhang, "Software Project Cost Estimation Based On Groupware", World Congress on Software Engineering, IEEE, 2009.

[2] Chen Qingzhang, Fang Shuojin, Wang Wenfu, "Development of the Decision Support System for Software Project Cost Estimation", World Congress on Software Engineering, IEEE, 2009.

[3] Y. F. Li, M. Xie, T. N. Goh, "A Study of Genetic Algorithm for Project Selection for Analogy Based Software Cost Estimation", IEEE, 2007.

[4] Yinhuan Zheng, Yilong Zheng, Beizhan Wang, Liang Shi, "Estimation of software projects effort based on function point", 4th International Conference on Computer Science and Education", 2009.

[5] Pichai Jodpimai, Peraphon Sophatsathit, and Chidchanok Lursin- sap, "Analysis of Effort Estimation based on Software Project Models", IEEE, 2009.

[6] Eduardo Aranha, Paulo Borba, "An Estimation Model for Test Execution Effort", First International Symposium on Empirical Software Engineering and Measurement, IEEE, 2007.

[7] Xiaochun Zhu, Bo Zhou, Li Hou, Junbo Chen, Lu Chen, "An Experience-Based Approach for Test Execution Effort Estimation", 9th International Conference for Young Computer Scientists, IEEE, 2008.

[8] Hao Wang, Fei Peng, Chao Zhang, Andrej Pietschker, "Software Project Level Estimation Model Framework based on Bayesian Belief Networks", Sixth International Conference on Quality Software (QSIC'06), IEEE, 2006.

[9] Yangyang Yu, Charlottesville, "A BBN Approach to Certifying the Reliability of COTS Software Systems", annual reliability and maintainability symposium, IEEE, 2003.

[10] Ying Wang, Michael Smith, “Release Date Prediction for Telecommunication Software Using Bayesian Belief Networks”, E Canadian Conference on Electrical and Computer Engineering, IEEE, 2002.

[11] Khaled Hamdan, Hazem El Khatib, Khaled Shuaib, “Practical Software Project Total Cost Estimation Methods”, MCIT 10, IEEE, 2010.

[12] JairusHihn, Hamid Habib-agahi, “Cost Estimation of Software Intensive Projects: A Survey of Current Practices”, IEEE, 2011.

[13] Khaled Hamdan, Mohamed Madi, “Software Project Effort: Different Methods of Estimation”, International Conference on Communications and Information Technology (ICCIT), Aqaba., IEEE, 2011.

[14] S. Bibi, I. Stamelos, L. Angelis, “Bayesian Belief Networks as a Software Productivity Estimation Tool, In proc. of 1st Balkan Conference in Informatics, pp. 585-596,November 2003

[15] Matthias Kerstner, “Software Test Effort estimation Methods”, 2 February 2011.

[16] Nancy Merlo Schett, “Seminar on software cost estimation”, University of Zurich, Switzerland, 2003.

[17] Chetan Nagar, “Software efforts estimation using Use Case Point approach by increasing technical complexity and experience factors”, IJCSE, ISSN:0975-3397, Vol.3 No.10 ,Pg No 3337- 3345,October 2011.

[18] Yunsik Ahn, Jungseok Suh,Seungryeol Kim, Hyunsoo Kim, “The software maintenance project effort estimation model based on function points”, Journal of software maintenance and evolution, 2003.

[19] Ashish Sharma, Dharmender Singh Kushwaha, “A Metric Suite for Early Estimation of Software Testing Effort using Requirement Engineering Document and its validation”, International Conference on Computer & Communication Technology (ICCCT), 2011

[20] Ricardo Britto, Emilia Mendes, Claes Wohlin, “A Specialized Global Software Engineering Taxonomy for Effort Estimation”, 2016 IEEE 11th International Conference on Global Software Engineering, IEEE, 2016

[21] Himani Rastogi, Swati Dhankar, Misha Kakkar, “A Survey on Software Effort Estimation Techniques”, IEEE, 2014

[22] Shashank Mouli Satapathy, Barada Prasanna Acharya, Santanu Kumar Rath, “Early stage software effort estimation using random forest technique based on use case points”, IET Softw., Vol. 10, Iss. 1, pp. 10–17, 2016

[23] Kirti Bareja, Abhishek Singhal, “A Review of Estimation techniques to reduce testing efforts in software development”, Fifth International Conference on Advanced Computing & Communication Technologies, 2015

[24] Soufiane Ezghari, Azeddine Zahi, Ali Idri, “A Learning Adaptation Cases Technique for Fuzzy Analogy-based Software Development Effort Estimation”, IEEE, 2014

[25] V. Anandhi, R. Manicka Chezian, “Regression techniques in software effort estimation using COCOMO dataset”, International Conference on Intelligent Computing Applications, 2014

[26] Shashank Mouli Satapathy, Mukesh Kumar, Santanu Kumar Rath, “Class Point Approach for Software Effort Estimation using Soft Computing Techniques”, IEEE, 2013

[27] Sumeet Kaur Sehra, Jasneet Kaur, Yadwinder Singh Brar, Navdeep Kaur, “Analysis of Data Mining Techniques for Software Effort Estimation”, 11th International Conference on Information Technology: New Generations, 2014

[28] Luigi Lavazza and Sandro Morasca, “Software Effort Estimation with a Generalized Robust Linear Regression Technique”, IEEE, 2012

[29] Petrônio L. Braga and Adriano L. I. Oliveira, Silvio R. L. Meira, “Software Effort Estimation using Machine Learning Techniques with Robust Confidence Intervals”, 19th IEEE International Conference on Tools with Artificial Intelligence, 2007

[30] S. Laqrichi, D. Gourc, F. Marmier, “Towards an effort estimation model for software projects integrating risk”, IEEE, 2008

[31] Jyoti G. Borade, Vikas R. Khalkar, “Software Project Effort and Cost Estimation Techniques”, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 8, August 2013

[32] Simon WU Iok Kuan, “Factors on Software effort estimation”, International Journal of Software Engineering & Applications (IJSEA), Vol.8, No.1, January 2017

[33] P.K. Suri, Pallavi Ranjan, “Comparative Analysis of Software Effort Estimation Techniques”, International Journal of Computer Applications (0975 – 8887) Volume 48– No.21, June 2012

[34] Sandeep Kad and Vinay Chopra, “Software Development Effort Estimation Using Soft Computing”, International Journal of Machine Learning and Computing, Vol. 2, No. 5, October 2012

[35] S.K. Mohanty and A.K. Bisoi, “Software Effort Estimation Approaches – A Review”, International Journal of Internet Computing ISSN No: 2231 – 6965, Volume 1, Issue 3, 2012

[36] S. Ramacharan and K. VenuGopala Rao, “Parametric Models for Effort Estimation for Global Software Development”, IEEE, Volume 1, No. 2, May 2013

[37] Pawandeep Kaur and Rupinder Singh, “A Proposed Framework for Software Effort Estimation Using the Combinational Approach of Fuzzy Logic and Neural Networks”, International Journal of Hybrid Information Technology Vol.8, No.10 , pp.73-80, 2015

[38] A. Khatibi Bardsiri, S. Mohsen Hashemi, “Software Effort Estimation: A Survey of Well-known Approaches”, International Journal of Computer Science Engineering, Vol. 3 No.01 Jan 2014

[39] Sivakumar D, Sureshkumar C, “Effort Estimation of Software Projects With Optimized Coefficients Using Soft Computing Technique”, IEEE Conference on Emerging Devices and Smart Systems (ICEDSS 2017) , 3-4 March 2017

[40] Jack H.C. Wu, Jacky W. Keung, “Utilizing Cluster Quality in Hierarchical Clustering for Analogy-Based Software Effort Estimation”, IEEE, 2017

[41] Mohamed Hosni and Ali Idri, Ali Bou Nassif, Alain Abran, "Heterogeneous Ensembles for Software Development Effort Estimation", 3rd International Conference on Soft Computing & Machine Intelligence, 2016

[42] Tirimula Rao Benala, Rohitha Bandarupalli, "Least Square Support Vector Machine in Analogy-based Software Development Effort Estimation", IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE-2016), December 23-25, 2016

[43] Samson Wanjala Munialo, Geoffrey Muchiri Muketha, "A Review of Agile Software Effort Estimation Methods", International Journal of Computer Applications Technology and Research Volume 5–Issue 9, 612-618, 2016

[44] JairusHihn, "COCOMO NASA 2 / Software cost estimation" [Online], Available: http://promise.site.uottawa.ca/SERepository/datasets/cocomonasa_2.arff. [Accessed: March 27, 2018]

List of Publications

[1] Nancy Sharma, Vineeta Bassi, "Comparative study of Software Effort Estimation Models", Proceeding of 3rdInternational Conference on Advancement in Engineering, Applied Science and Management (ICAEASM 2018), pp. 35-40, May 2018

Plagiarism Report of Thesis

ORIGINALITY REPORT

Nancy Sharma
Nivedita Bisht

9%	6%	5%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	dspace.thapar.edu:8080 Internet Source	1%
2	Submitted to Panjab University Student Paper	1%
3	gdeepak.com Internet Source	<1%
4	ethesis.nitrkl.ac.in Internet Source	<1%
5	www.ijarcsse.com Internet Source	<1%
6	Tirimula Rao Benala, Rohitha Bandarupalli. "Least Square Support Vector Machine in Analogy-Based software development effort estimation", 2016 International Conference on Recent Advances and Innovations in Engineering (ICRAIE), 2016 Publication	<1%
7	methodsandtools.com Internet Source	<1%

8	SubmittedtoChandigarhUniversity StudentPaper	<1%
9	www.gdeepak.com InternetSource	<1%
10	www.growingscience.com InternetSource	<1%
11	ijcsit.com InternetSource	<1%
12	vbn.aau.dk InternetSource	<1%
13	PetronioL.Braga,AdrianoL.I.Oliveira,Silvio R.L.Meira."SoftwareEffortEstimationUsingMachineLearningTechniqueswithRobustConfidenceIntervals",19thIEEEInternationalConferenceonToolswithArtificialIntelligence(ICTAI2007),2007 Publication	<1%
14	SubmittedtoMuchinCollegePrep StudentPaper	<1%
15	Kyung-SuKim."Supplementarylossconcealmenttechniqueforimagetransmissionthroughdatahiding", MultimediaToolsandApplications,03/17/2009 Publication	<1%
16	JackH.C.Wu,JackyW.Keung."Utilizing	

clusterqualityinhierarchicalclusteringforanalogy-basedsoftwareeffortestimation",20178thIEEEInternationalConferenceonSoftwareEngineeringandServiceScience(ICSESS),2017
Publication

<1%

17 SoftwareProjectEffortEstimation,2014.
Publication

<1%

18 Lavazza,L.,andS.Morasca."Softwareefforteestimationwithageneralizedrobustlinearregressiontechnique",16thInternationalConferenceonEvaluation&AssessmentinSoftwareEngineering(EASE2012),2012.
Publication

<1%

19 SubmittedtoSavitribaiPhulePuneUniversity
StudentPaper

<1%

20 MohamedHosni,Alildri,AliBouNassif,AlainAbran."HeterogeneousEnsemblesforSoftwareDevelopmentEffortEstimation",20163rdInternationalConferenceonSoftComputing&MachineIntelligence(ISCMI),2016
Publication

<1%

21 SubmittedtoTobbUniversityofEconomics&Technology
StudentPaper

<1%

D.Sivakumar,C.Sureshkumar."Effort

22	estimationofsoftwareprojectswithoptimizedcoefficientsusingsoftcomputingtechnique",2017ConferenceonEmergingDevicesandSmartSystems(ICEDSS),2017 Publication	<1%
23	research.ijcaonline.org InternetSource	<1%
24	TirimulaRaoBenala."GeneticAlgorithmforOptimizingNeuralNetworkBasedSoftwareCostEstimation",LectureNotesinComputerScience,2011 Publication	<1%
25	Sun-JenHuang."Applyingfuzzyneuralnetworktoestimatesoftwaredevelopmenteffort",AppliedIntelligence,04/2009 Publication	<1%
26	SubmittedtoPunjabTechnicalUniversity StudentPaper	<1%
27	SubmittedtoThaparUniversity,Patiala StudentPaper	<1%
28	DanielMcInerney."AcomparativeanalysisofkNNanddecisiontreemethodsfortheIrishNationalForestInventory",InternationalJournalofRemoteSensing,10/2009 Publication	<1%
29	SubmittedtoUniversityof Walescentral	

	institutions StudentPaper	<1%
30	SubmittedtoUniversityofAuckland StudentPaper	<1%
31	SubmittedtoUniversityofWales,Bangor StudentPaper	<1%
32	SubmittedtoHigherEducationCommissionPakistan StudentPaper	<1%
33	SubmittedtoSouthernNewHampshireUniversity-DistanceEducation StudentPaper	<1%
34	www.computer.org InternetSource	<1%
35	nvl.nist.gov InternetSource	<1%
36	dblp.dagstuhl.de InternetSource	<1%
37	www.dcc.ufla.br InternetSource	<1%
38	SMNadgeri,VPHulsure,ADGawande."ComparativeStudyofVariousRegressionMethodsforSoftwareEffortEstimation",20103rdInternationalConferenceonEmergingTrendsinEngineeringandTechnology,2010	<1%

	Publication	
39	qmro.qmul.ac.uk InternetSource	<1%
40	Qin,Xiaotie, and MiaoFang."SummarizationofSoftwareCostEstimation",ProcediaEngineering,2011. Publication	<1%
41	www.apjmr.com InternetSource	<1%
42	SubmittedtoVirginiaInternationalUniversity StudentPaper	<1%
43	Srivastava,PraveenRanjan."Optimal SoftwareReleaseUsingTimeandCostBenefitsviaFuzzyMulti-CriteriaandFaultTolerance",JournalofInformationProcessingSystems,2012. Publication	<1%
44	www.ijarcce.com InternetSource	<1%
45	toc.proceedings.com InternetSource	<1%
46	SoufianeEzghari,AzeddineZahi,Alildri."Alearningadaptationcasestechniqueforfuzzyanalogy-basedsoftwaredevelopmenteffortestimation",2014SecondWorldConferenceon	<1%

ComplexSystems(WCCS),2014
Publication

47	ijarcsse.com InternetSource	<1%
48	digital-library.theiet.org InternetSource	<1%
49	etd.lib.metu.edu.tr InternetSource	<1%
50	scholarcommons.usf.edu InternetSource	<1%
51	ZhuXiaoliang,YanHongcan,WangJian,WuShangzhuo."Researchandapplicationoftheimprovedalgorithm C4.5onDecisiontree",2009InternationalConferenceonTestandMeasurement,2009 Publication	<1%
52	Bo-SukYang."Randomforestsclassifierformachinefaultdiagnosis",JournalofMechanicalScienceandTechnology,09/2008 Publication	<1%
53	www.sersc.org InternetSource	<1%
54	Idri,Ali,FatimaazzahraAmazal,andAlainAbran."AccuracyComparisonofAnalogy-BasedSoftwareDevelopmentEffort	<1%

EstimationTechniques:SOFTWAREDEVELOPMENTEFFORTESTIMATION TECHNIQUES",InternationalJournalofIntelligent Systems, 2015.
Publication

---

55 Jorgensen, Magne, andSteinGrimstad."TheImpactof IrrelevantandMisleadingInformationonSoftwareDevelopmentEffortEstimates:ARandomizedControlledFieldExperiment",IEEETransactionsonSoftwareEngineering,2011. <1%
Publication

---

56 MohamedHosni,Alildri,AlainAbran."Investigatingheterogeneousensembleswithfilterfeatureselectionforsoftwareeffortestimation",Proceedingsofthe27thInternationalWorkshoponSoftwareMeasurementand12thInternationalConferenceonSoftwareProcessandProductMeasurementon-IWSMMensura'17,2017 <1%
Publication

---

57 Chiu,N.H.."Theadjustedanalogy-basedsoftwareeffortestimationbasedonsimilaritydistances",TheJournalofSystems&Software,200704 <1%
Publication

---

58 dl.acm.org <1%
InternetSource

59	V.Resmi,S.Vijayalakshmi,R.SubashChandrabose."Aneffectivesoftwareprojecteffortestimationsystemusingoptimalfireflyalgorithm",ClusterComputing,2017 Publication	<1%
60	Li,Y.F.."Adaptiveridgeregressionsystemforsoftwarecostestimatingonmulti-collineardatasets",TheJournalofSystems&Software,201011 Publication	<1%
61	K.Usharani,V.VignarajAnanth,D.Velmurugan."Asurveyonsoftwareeffortestimation",2016InternationalConferenceonElectrical,Electronics,andOptimizationTechniques(ICEEOT),2016 Publication	<1%
62	ijsce.org InternetSource	<1%
63	infocomp.dcc.ufla.br InternetSource	<1%
64	imtr.ircam.fr InternetSource	<1%
65	Seyagh,Maria,MazouzElMostapha,AbdellahJarid,DrissCherqaoui,AndreeaSchmitzer,andDidierVillemin."Patternrecognition:ApplicationofSupportVectorMachines,	<1%

ArtificialNeuralNetworksandDecisionTreesfor anti-HIVactivitypredictionoforganiccompounds",2011InternationalConferenceonMultimediaComputingandSystems,2011.
Publication

66 YunsikAhn."Thesoftwaremaintenanceprojectefforteestimationmodelbasedonfunctionpoints",JournalofSoftwareMaintenanceandEvolutionResearchandPractice,03/2003
Publication <1%

67 Eugenio Capra,ChiaraFrancalanci,FrancescoMerlo."TheEconomicsofOpenSourceSoftware:AnEmpiricalAnalysisofMaintenanceCosts",2007IEEEInternationalConferenceonSoftwareMaintenance,2007
Publication <1%

68 www.ijritcc.org
InternetSource <1%

69 ieeexplore.ieee.org
InternetSource

Excludequotes	Off	Excludematches	<8words
Excludebibliography	On		