ISSN:0975-9646

Predicting Error in Software Effort Estimate using K-means and Quad Tree algorithm

Rutvi Vanapalli, Ch.Satyananda Reddy

Department of Computer Science and Systems Engineering, Andhra University
Andhra Pradesh, India

***Abstract*: software estimation plays a vital role in the industry. Parameters like cost, effort and schedule are estimated using various industry standard methods. But there is no defined model to predict the estimation of effort during the initial stages of the project. In this paper, a new method based on data mining concepts is defined to help determine the error between the estimated effort and the actual effort, in the initial stages of the project. This method is based on Quad-tree and K-means algorithms. The results of this model are compared against an industry standard model- the intermediate COCOMO model.**

***Keywords*: Software Effort estimation, data-mining, K-means algorithm, Quad-tree algorithm, COCOMO model, fault prediction, error.**

I. Introduction

"Estimation" is a basic activity of every project and plays a vital role in the industry today. Every project has a set of goals, and the process of estimation gives an accurate idea of what is required and what is to be done, in terms of time, cost and other resources. Resources are always limited and hence an accurate estimate is very important in order to decide if the goals can be achieved, they are also used for creating and updating project plans. But, the software industry has limited models to help them predict parameters like cost, effort and schedule.

And this is partly because it is quite difficult to estimate software. Factors like individual's performance and productivity causes a major difference, software being intangible and that there might be changes that need to be done during the life cycle of the project, can all affect the overall estimated factor.

There are many industry standard methods like Putnam's SLIM model and COCOMO model. However, there are certain limitations such as: Firstly, model structure: The size of the product is a key in many models to determine the effort required in building products. Secondly, overly complex models: The organization's characteristics have a major influence on its productivity. Lastly, product size estimation: Product size is required for every model as an input, but this cannot be measured during the early stages of the project lifecycle.

In order to predict error in effort, that is the difference between the estimated effort and actual effort, right during the initial stages of the project lifecycle, a new method using data mining concepts K-means algorithm and Quad-tree algorithm is proposed in this paper.

The organization of the paper is as follows: Section 2 states the state-of-art work in this field of study; Section 3 contains the proposed method in detail. An analysis with respect to intermediate COCOMO model is stated in section 4. The results are discussed in section 5.

II. Related Work

The state-of-art in the field of software effort estimation using data mining concepts is reviewed

Seliya et al. [1] proposed a constraint-based semi-supervised clustering scheme that uses K-means clustering method as the underlying clustering algorithm for this problem. Catal et al.[4] proposed a clustering and metrics threshold based software fault prediction approach and explored it on three datasets. Meenakshi et al. [6] propose the usage of Quad tree and Expectation maximization algorithm which is also a clustering algorithm known for iterative refinement. P.S. Bishnu and V.Bhattacharjee[3] determine software faults using Quad Tree and K-means algorithm and the Quad Tree used here is a simple region quad tree where the child nodes are only split into four.

III. Proposed Method

This model uses the data-mining partitioning algorithm known as the K-means algorithm, which is a data-mining clustering algorithm, along with the Quad-Tree algorithm which is used to feed the input data centres to the data mining conceptual model.

The advantage of using this model over the pre-existing models is that, this model does not depend on the Lines of Code attribute. This model can also be applied over various model structures, where the values of "a" and "b" used in the effort equation have no significance as it is not based on the effort formula. It can be applied on multiple types of projects where the concept of "reuse" is implemented, thereby, overcoming majority of the shortcomings found in the other models.

The proposed model can be categorized under the "Cost model" of Effort estimation models as it based on 14 cost drivers and also depends on one other attribute known as "classes" which is simply the categorization class of the project based on its type. The model is based on an application, generally used for predicting the Software Faults in program modules.

This model uses the project datasets available in PROMISE NASA Datasets.

A. process summary

The model is based on data mining concepts. This method implements the K-means algorithm which is a data mining partitioning algorithm used to partition and group similar data sets. This is based on a distance metric. It simply calculates the distance of an input point mapped in a two-dimensional structure against the calculated midpoints. The distance must satisfy the given threshold distance and those having similar values are mapped and grouped together forming clusters.

However, this algorithm has the following drawbacks:

- The user must identify the initial number of clusters.
- The selection of the data centres must be done manually.
- It is sensitive to noise.

Hence, an algorithm known as the Quad-tree algorithm is implemented, which eradicates these drawbacks.

The Quad-tree algorithm is first applied for finding initial cluster centres for the K-means algorithm. The Quad tree is a tree data structure in which the internal nodes have exactly four children. These trees are divided into two-dimensional spaces which are further subdivided into four quadrants known as the "sub-quadrants" where each leaf node contains some data corresponding to a specific sub-region.

A quad-tree with a depth of "n" maybe used to represent the input datasets of 2^n x 2^n pixels, where each pixel value is 0 or 1. The root node represents the entire data region.

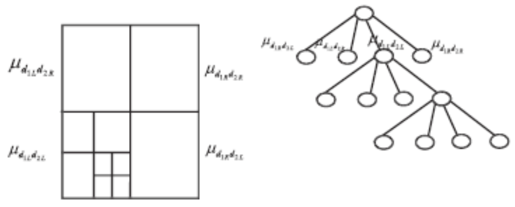

Fig 3.1: Quad Tree implementation

The input data sets for K-means and Quad tree on which the data mining concepts are demonstrated and applied is taken from PROMISE dataset of NASA which is an open repository dataset source.

The PROMISE dataset contains 60 project data that can be used for software cost estimation using the various industry standard models. The database consists an overall of 17 effort multipliers which vary in the range of Very_low to Extreme_High, an LOC attribute and also one goal field.

The effort multipliers fall into 3 groups- those that are positively correlated to more effort, those that are negatively correlated to more effort and the third group which just contains the "sced" effort multiplier- schedule information, has a U-shaped correlation to effort. However, the sced effort multiplier and the LOC attribute are not considered in this model because the LOC attribute cannot be actually predicted in the early stages of the project and sced is u-shaped correlated.

Out of the 17 effort multipliers, only 14 are used in this project along with the attribute "Classes" which states the categorization of projects based on the type of project and this value ranges from 1-4.

The Effort multipliers used in this project are as follows:

RELY: Required software reliability.
DATA: Database size.
CPLX: Process complexity.
TIME: Time constraint for CPU.
STOR: Main memory constraint.
VIRT: Machine volatility.
TURN: Turnaround time.
ACAP: Analysts capability.
AXEP: Application experience.
PCAP: Programmers capability.
VEXP: Virtual machine experience.
LEXP: Language experience.
MODP: Modern programming practises.
TOOL: use of software tools.

These 15 attributes (14 effort multipliers+ classes) are first represented on the desktop application frontend on being retrieved from the database. Out of which, the 14 effort multipliers are divided into X and Y attributes as follows:

X-attribute: Those effort multipliers that need to be decreased to decrease the effort. They are- rely, cplx, stor, virt, data, time and turn.

Y-attribute: Those effort multipliers that are to be increased to decrease effort. They are- acap, pcap, aexp, modp, tool, vexp and lexp.

The class attribute is not included in either of these two mappings.

On grouping the data into (X, Y) tuple datasets, the Quad-tree algorithm is applied on these to pick out the initial data centres.

The midpoints are first determined by applying the general mathematical mid-point formula:

$$D_1 = \frac{x1+x2+x3+x4+......+xn}{2}$$

$$D_2 = \frac{y1+y2+y3+y4......+yn}{2}$$

After the midpoints are obtained, each of the (X_n, Yn) tuple is compared against the input data tuples and a Quad-tree is hence formed based on the following four conditions:

- If $X_n>D_1$ and $Y_n>D_2$, then place this tuple in the first sub-quadrant.
- If $X_n<=D_1$ and $Y_n>D_2$, then place this tuple in the second sub-quadrant.
- If $X_n<=D_1$ and $Y_n<=D_n$, then place this tuple in the third sub-quadrant.
- If $X_n>D_1$ and $Y_n<=D_n$, then place this tuple in the fourth sub-quadrant.

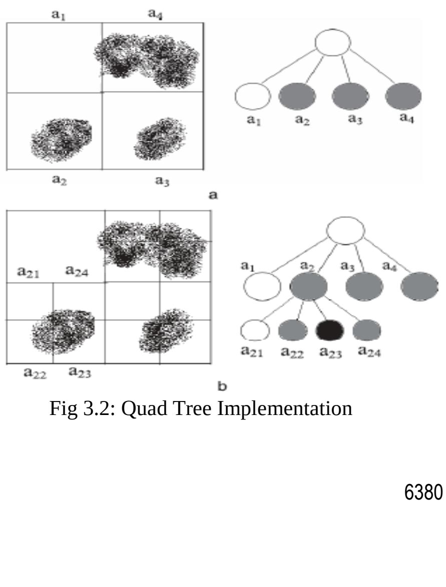

Fig 3.2: Quad Tree Implementation

This is an iterative process and a quad-tree is formed. Those nodes or sub-quadrants that further need to be split into sub-quadrants, that is, those nodes having a value of more than 0 or 1, are chosen as black nodes. The nodes having the value 0 are known as grey nodes and those nodes having the value 1 are known as white nodes. Black nodes are those nodes that have a value more than that of the threshold. These black nodes are considered to be the input initial data centre points for the K-means algorithm.

Once the initial data clusters are determined, the K-means algorithm is applied on the input datasets which still exist in the form of a two-dimensional structure.

The K-means algorithm calculates the Euclidean distance between the data centre to each of the input tuple data sets and thereby classifies them into clusters.

Euclidean distance= $\sqrt{((x_2 - x_1)^2 + (y_2 - y_1)^2)}$

The number of clusters thus formed depends on the initial number of cluster centres detected by the quad-tree algorithm.

Having the input data clustered the software fault data is predicted in terms of false positive ratio (FPR), false negative ratio (FNR) and the error rate.

FPR=$\frac{B}{A+B}$ FNR=$\frac{C}{C+D}$ Error=$\frac{B+C}{A+B+C+D}$

Where A is the fault labels which are actually faults, B is the fault labels that are predicted to be non-faults, C is the fault labels that are non-faulty but predicted as faults, D is the fault labels that are actually non-faulty.

Table 3.1: Confusion matrix

	Predicted labels		
		False(Non-Faulty)	**True(Faulty)**
Actual Labels	**False(Non-Faulty)**	A(True negative)	B(False positive)
	True(Faulty)	C(False negative)	D(True positive)

The error obtained here is the difference in effort estimation that is between the actual estimate that is derived at the end of the project upon project delivery and to that of the estimated effort done during the initial stages of the project.

B. Algorithm

Quad Tree- K-means algorithm

Step 1: Read the fifteen attribute values- Rely, Cplx, Data, Time, Stor, Virt, Turn, Acap, Axep, Pcap, Vexp, Lexp, Modp, Tool and classes.

Step 2: Divide these fifteen attributes into two-dimensional data.

Xaxis: Rely+Data+Cplx+Time+Stor+Virt+Turn

Y-axis: Acap+Aexp+Pcap+Vexp+Lexp+Modp+Tool.

Step 3: Calculate the midpoint of X-axis and Y-axis points respectively by using the mathematical general formula.

$$D_1 = \frac{x1+x2+x3+x4+......+xn}{2}$$

$$D_2 = \frac{y1+y2+y3+y4......+yn}{2}$$

Step 4: Quad tree algorithm is implemented in this step. Every tuple (x, y) data point is compared against the D1 and D2 midpoints. The tuples are classified into four children based on the following conditions:

- If $X_n > D_1$ and $Y_n > D_2$, then place this tuple in the first sub-quadrant.
- If $X_n <= D_1$ and $Y_n > D_2$, then place this tuple in the second sub-quadrant.
- If $X_n <= D_1$ and $Y_n <= D_n$, then place this tuple in the third sub-quadrant.
- If $X_n > D_1$ and $Y_n <= D_n$, then place this tuple in the fourth sub-quadrant.

Step 5: The child nodes in the Quad tree having value =2, that is, more than 0 or 1, are termed as Black nodes whose size is greater than the threshold.

Step 6: All black nodes found in Quad tree become the initial data centres for the K-means algorithm.

Step 7: All the (x, y) tuples are compared against the data centres using Euclidean distance formula, and respective data clusters are formed.

Euclidean distance= Euclidean distance= $\sqrt{((x_2 - x_1)^2 + (y_2 - y_1)^2)}$

Step 8: False positive ratio, false negative ration and the error are calculated using the formulae

FPR=$\frac{B}{A+B}$ FNR=$\frac{C}{C+D}$ Error=$\frac{B+C}{A+B+C+D}$

IV. RESULT ANALYSIS

The results are compared against the industry standard-intermediate COCOMO model.

The intermediate COCOMO model is based on all 15 effort multipliers including the Lines of Code (LOC) metric. Using these effort multipliers, the Effort Adjustment Factor (EAF) is determined.

EAF= $EM_1 * EM_2 .... * EM_{13} * EM_{14}$

The estimated Effort is then calculated using the general formula:

Estimated Effort= a $(KLOC)^b$EAF

Where a=3.2, b=1.05 and KLOC= LOC/1000

The error in effort using the Intermediate COCOMO model can be calculated using the following formula:

Error in Effort= $\frac{\text{ESTIMATED EFFORT} \sim \text{ACTUAL EFFORT}}{\text{ACTUAL EFFORT}}$

The values of error in effort estimate using the proposed method-K-means Quad tree algorithm are compared against the Intermediate COCOMO model outputs and the results are tabulated in table 4.

Table 4.1: Results QDK Vs COCOMO

Serial No.	No. Of input sets	QDK value	COCOMO value
1.	1	0.000	0.007
2.	3	0.5	0.47
3.	10	0.3	0.5
4.	20	0.5	0.36
5.	31	0.5	0.38
6.	45	0.34	0.39
7.	51	0.32	0.40

V. Conclusion

The goal of this paper is to propose a new method to determine the difference in Effort estimated and the Actual Effort, right in the initial stages just basing on the Effort Multiplier data using K-means clustering algorithm and Quad tree algorithm.

The K-means clustering algorithm is used to partition and cluster the input effort multiplier data that consists of 15 attributes that are taken from the PROMISE dataset, which is available online. The partitioning algorithm needs some initial data centres to be chosen and this is done by using the Quad Tree algorithm technique which is basically a tree-structure that splits in a 4-way branching system. Once the initial cluster centres are recognized, the data is divided into faulty and non-faulty data and the Error in Effort Estimate is hence predicted using data-mining techniques.

References

[1] N. Seliya and T.M. Khoshgoftaar, "Software Quality Classification Modeling Using the PRINT Decision Algorithm," Proc. IEEE 14th Int'l Conf. Tools with Artificial Intelligence, pp. 365-374, and 2002.

[2] http://archive.ics.uci.edu/ml/datasets/Iris, 2012.

[3] P.S. Bishnu and V. Bhattacharjee, "A New Initialization Method for KMeans Algorithm Using Quad Tree," Proc. Nat'l Conf. Methods and Models in Computing (NCM2C), pp. 73-81, 2008.

[4] C.Catal, U.Sevim and B.Diri "Software Fault prediction of unlabeled program modules" Proc. Of the world congress on Engineering 2009 Vol I WCE 2009, July 1-3, 2009, ISBN: 978-988-17012-5-1

[5] Promise data sets-NASA

[6] Intermediate COCOMO model-Wikipedia

[7] Effort estimation-Wikipedia

[8] Textbook on Software metrics: A rigorous and practical approach by Norman E. Fenton and Shari Lawrence Pfleeger.

[9] Meenakshi PC, Meenu S, Mithra M and Leela Rani P. " Fault Prediction using Quad Tree and Expectation Maximization Algorithm". International Journal of Applied Information Systems 2(4):36-40, May 2012. Published by Foundation of Computer Science, New York, USA