Improvement in Accuracy of Software Estimation using T3 an Improved Decision Tree Classifier

Miss Sumera W.Ahmad Asst. Prof
Department of CSE PRMIT&R Badnera , Amravati Maharashtra
Email: sumeraahmad2003@gmail.com

Dr.G.R.Bamnote Professor Department of CSE PRMIT&R Badnera, Amravati, Maharashtra
Email: grbamnote@rediffmail.com

Abstract

Software estimation data from the past history of the projects are valuable resources from which accurate and potential useful knowledge can discovered by using data mining. Data mining can assist and support the software engineer for accurate cost estimation of project and investigate research. The objectives of this paper is to propose a method for building accurate predictive estimation schedule based on past experience of the software estimation data to improve the accuracy. The paper reports an experience from the development and use of different software estimation models .The model COCOMO, SEER, SIM were developed by researchers and accuracy was measured .This paper suggest the use of T3 classification algorithm on classification of past software estimation data .It also compare this method with other well established data mining methods and identify the strengths and weaknesses.

Introduction

Today many industries generate and collect huge amount of data which are traditionally used to analyze manually. However many unrevealed and potentially useful relationships may not be identified by analyst. This tremendous growth of data requires an automated way to extract useful parameter. Data mining results in discovering new interesting and useful knowledge and patterns from database. The discovered knowledge can be applied in corresponding domain to increase working efficient and improve quality of decision making. Data mining application have already been proven to provide benefit to many areas of software engineering including error detection, fault finding, and software optimization. Classification is a data mining technique which addresses the problem of constructing a predictive model for estimation in software engineering through examples of records with known class decision tree are the most well establish classification methods when it comes to selecting a decision tree classifier the task is not an easy one as clear winners simply does not exist. Important thing is to select the classifier generalization, accuracy and tree size .In this paper T3 describe an improved version of existing algorithm.T3 builds decision tree of depth at most three .T3 builds tree larger than T2 and adopts a less greedy approach in tree building phase. This is demonstrated using a comparative study based on real software repository.

The remainder of the paper is organized as follows: In section 2 we discuss the problem of accurate estimation of software project in the industry. Section 3 gives a brief description of C4.5, a known data mining classification algorithm and T3 which improves the tree building process; Comparison and evaluation of performance of two algorithm is done in Section 4, conclusion and future work are presented in section 5.

2. Description of the Problem Domain

Cost estimation is a process or an approximation of the probable cost of software program, or a project computed on the basis of available information. Accurate cost estimation is very important for every kind of project, if the project is not estimated in the proper way it results the cost of the project very high sometimes it may reach 150-200% more than the original cost. So it is very necessary to estimate the project correctly.

Software cost estimation is related to how long and how many people are required to complete a software project. Software cost estimation starts at the proposal state and continues throughout the life time of a project. The estimation includes project size estimation, effort estimation, developing initial project schedules and finally estimating overall cost of the project. Software development has become an essential question because many projects are still not completed on schedule with under or over estimation of efforts leading to their own particular problems. Therefore, in order to manage budget and schedule of software projects researchers have developed various software cost estimation models and also measured those models .
Accurate software cost estimates are critical to both developers and customers. They can be used for generating request for proposals, contract negotiations, scheduling, monitoring and control. The estimation is a process of determining amount of efforts, money, resources and time for building a software project with the help of available past history of the projects data. Many estimation methods have been proposed in last 30 years and almost all methods require quantitative information of productivity, size of project and other important factors that affect the project. The following reason makes accurate cost estimation important.

1) It helps to classify and prioritize development projects with respect to an overall business plan.
2) It can be used to determine what resources to commit to the project and how well these resources will be used.
3) It can be used to assess the impact of changes and support preplanning.
4) Projects can be easier to manage and control when resources are better matched to real needs.
5) Customers expect actual development costs to be in line with estimated costs.
6) Software cost estimation historically has been a major difficulty in software development.
7) A software cost estimate done early in the project life cycle is generally based on less precise inputs and preliminary design specifications and their relationships are not well understood.
8) Lack of a historical database of cost measurement that means historical

data is sometimes incomplete, inconsistent or inaccurate.

9) Lack of trained estimators and estimators with the necessary expertise.
10) Software is intangible, invisible, and intractable so it is more difficult to understand and estimate a product or process that cannot be seen and touched.
11) Too low effort estimates may lead to project management problems, delayed deliveries, budget overruns and low software quality, too high effort estimates may lead to lost business opportunities and inefficient use of resources.

3.Classification Algorithm Description

The aim behind the experimentation was that some of the attributes might be predictive of class attributes. Building classification trees useful method suitable for this study. There are many algorithm that produces such tree C4.5 and SLIQ are reported to be the best performing in this area.T3 is an algorithm that results in small but accurate data sets. Thus a comparison is made using C4.5 and T3 the main feature are described below.

C4.5: It is the well known classification algorithm that adopts a depth first strategy (divide and conquer) using gain ratio, as a split criterion ,resulting in reduced bias in favor of many valued attributes ,as compared to algorithm that use the information gain. The algorithm however, uses also considered to be biased in favor of continuous attributes, this being the reason for improvement .Splits in continuous values are binary, dividing the search space into two disjoint parts. Unknown values are ignored in the training phase resulting in classification errors. The probable result is estimated in the case of unknowns during testing .C4.5 employs a pruning technique that replaces subtree with leaves, thus reducing over fitting. The accuracy achieved is high as compared to other algorithm.

T3 is an algorithm base on T2.T3 calculate optimal decision tree up to depth 3 by using two kinds of decision splits on nodes. Discrete splits are used on discrete attributes where the node has as many branches as there as possible attributes values. Internal splits are used on continuous attributes where a node has as many brunches as there are intervals. The number of intervals is restricted to be either at most as many as the number of existing classes plus one ,if all the branches of the decision node lead to leaves, are to be at most 2 otherwise. The attribute value “unknown” is treated as a special attribute value. Each decision mode (discrete or continuous) has one additional branch, which takes care of unknown attributes values. In fact this way of treating is unknown attribute is reported to perform better than that of C4.5. In the tree building Phase at each node, all attributes are examined in order to select one on which split will be performed for the node, When attribute is discrete, the relative frequency of all of its possible values are calculated. For continuous attributes the same approach would be inefficient because of the number of possible values and the resulting law frequencies of them. For that reason local discretion is used. Finally a split is performed on an attribute if it results in maximum accuracy.

4. Comparison and Evaluation of Performance of Two Algorithm

Consequently T3 produces a tree structure, which determines how important is an attributes in the classification process in comparison to C4.5 which uses gain ratio. To carry out this local discretion of continuous attributes its values have to be partitioned into multiple the set of intervals that minimize the classification error is framed by through exploration instead of heuristically applying recursive binary splitting .The search for these intervals is computational expensive. So T3 restricts decision tree to three levels of tests, where only the third level of employs non binary splits of continuous attributes.

T3 does not use pruning technique instead it uses a parameter (Maximum Acceptable Error).It is a positive real number less than 1 used as a stopping criterion during building tree. T2 uses a greedy approach when designing the tree, thus further splitting at a node would stop only if the records already classified in this node, belonged to a single class. However, this greedy approach is not optimal, because reducing the error in the leaf nodes does not necessarily result in minimizing the overall error in the whole tree. It has proved that a strategy choosing locally optimal splits necessarily produces sub-optimal trees. It should be noted that classification error show how many instances of a training set have been incorrectly classified, while generalization error indicates how many instances of a testing set have been incorrectly classified.

MAE allows the user to specify the level of purity in the leaves and to stop further building of the tree at a potential node split. Setting MAE to have 4 distinct values, namely 0.0, 0.1, 0.2 and 0.3, meaning that splitting at a node stops even if the error in that node is equal to or below a threshold of 0, 10, 20 or 30 per cent respectively.

More precisely, building the tree would stop at a node in two cases. In the first case, building stops when the maximum depth is reached, i.e. 3 when T3 is used or 2 when T2 is used. In the second case, building stops at that node only when all the records remaining there to be classified belong to the same class in a minimum proportion of 70, 80, 90 or 100%. Proposing to use eight different versions of T3, four with maximum depth two and MAE set to 0, 0.1, 0.2 and 0.3 respectively.

Conclusion

A propose algorithm T3 an improved decision tree classifier a method for building accurate predictive estimation schedule based on past experience of the software estimation data to get a improved and the accurate estimation would be better when experimented with real data from the software engineering repository.T3 could demonstrate higher predictive ability as it will be demonstrated by the results in near future . Finally, a user-friendly interface that would suggest which version is to be used depending on the nature of the data and the task in hand would enhance the systems performance.

References

1 Kamber, M., Winstone, L., Gong,

W., Cheng, S., Han, J. Generalisation and Decision Tree Induction: Efficient Classification in Data Mining. In Proc. Int'l Workshop Research Issues Data Engineering (RIDE'97) 1997; 111-120.

2 Ganti, V., Gehrke, J., Ramakrishnan, R. Mining Very Large Databases. IEEE Computer, Special issue on Data Mining, 1999; 32(8):38 – 45.

3 Tjortjis, C., Keane J.A, T3: an Improved Classification Algorithm for Data Mining. Lecture Notes Computer Science Series, Springer-Verlag, Vol. 2412, 2002; pp. 50-55.

4 Mehta, M., Agrawal R., Rissanen, J. SLIQ: A Fast Scalable Classifier for Data Mining. In Proc. 5th Int'l Conf. Extending Database Technology 1996; 18-32.

5 Hand, D.J., Mannila, H. and Smyth, P. Principles of Data Mining. The MIT Press, 2001; 21.

6 Murthy S., Saltzberg S. Decision Tree Induction: How effective is the Greedy Heuristic? In Proc. 1st Int'l Conf. KDD and DM 1995; 156-161.

7 Quinlan, J. R. Unknown attribute values in induction. In Proc. 6th Int’l Machine Learning Workshop 1989; 164-8.

8 Quinlan, J.R. Improved Use of Continuous Attributes in C4.5, Journal AI Research 1996; 4: 77-90.

.