Available online at www.sciencedirect.com
ScienceDirect
Information and Software Technology 50 (2008) 656–669
INFORMATION AND SOFTWARE TECHNOLOGY
www.elsevier.com/locate/infsof

Combining probabilistic models for explanatory productivity estimation

S. Bibi *, I. Stamelos, L. Angelis

Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki 54124, Greece

Received 15 May 2006; received in revised form 20 June 2007; accepted 27 June 2007
Available online 6 July 2007

Abstract

In this paper Association Rules (AR) and Classification and Regression Trees (CART) are combined in order to deliver an effective conceptual estimation framework. AR descriptive nature is exploited by identifying logical associations between project attributes and the required effort for the development of the project. CART method on the other hand has the benefit of acquiring general knowledge from specific examples of projects and is able to provide estimates for all possible projects. The particular methods have the ability of learning and modelling associations in data and hence they can be used to describe complex relationships in software cost data sets that are not immediately apparent. Potential benefits of combining these probabilistic methods involve the ability of the final model to reveal the way in which particular attributes can increase or decrease productivity and the fact that such assumptions vary among different ranges of productivity values. Experimental results on two data sets indicate efficient overall performance of the suggested integrated method.

Keywords: Software cost estimation; Machine learning; Association Rules; Classification and Regression Trees

1. Introduction

Mature software organizations collect data regarding software development in hope of extracting useful information from them and thus understanding better their processes and products. Software management data, such as the size of programs, the effort required for their completion, the tools, packages and methodologies utilized, contain a wealth of information about a project status, progress and evolution. Using well-established data mining techniques, practitioners and researchers can explore the potential of this data by extracting patterns and identifying relations among software project attributes. Such information can be ideally exploited to describe and evaluate current software projects and furthermore, to predict future behaviour by estimating values of certain attributes of projects under development, based on some other known variables.

Cost estimation in software engineering is the process of predicting the amount of effort or productivity required for the completion of a software artefact. Typically, software cost estimation involves initially an assessment on the project attributes and then the application of a method for the generation of an estimate.

One of the most usual cost estimation approaches is the construction of a conceptual model involving a set of variables and a set of logical and quantitative relationships between them. The construction and the training of such a model is based on all of the available information contained in data sets.

The conceptual estimate resulting from a model will be finally interpreted, probably by a software manager who will attempt to confirm the estimate intuitively. It is important for the end-user to understand the rationale under which the estimate is performed in order to trust and adopt the estimate. Therefore, software cost estimation models should also have an explanatory usefulness in order to convince the estimator for the predicted values obtained. Accuracy and explanatory value of models are both

* Corresponding author. Tel.: +30 2310991910.
E-mail addresses: sbibi@csd.auth.gr (S. Bibi), stamelos@csd.auth.gr (I. Stamelos), lef@csd.auth.gr (L. Angelis).

0950-5849/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.infsof.2007.06.004

important factors that will ensure interaction between the conceptual cost estimation model and the final estimator.

The accuracy of a software cost model is its ability to predict the effort or the productivity value of a new project closely to the actual one. Explanatory value of a model is the ability of the model to describe and provide reasoning of the events and relations that produced a certain estimate in an interpretable and intuitive form.

Many studies have been conducted so far in software cost estimation, e.g. [14,20,27,36], focusing mainly in the estimation accuracy of cost models. One study [27] that compared estimation models in terms of explanatory value, reached the conclusion that in the particular data set the estimation methods with explanatory value such as rule induction and Case Base Reasoning (CBR) were less accurate than Neural Nets [37] whose output is difficult to understand and interpret. In general, accurate methods such as regression models [15,36] may produce results that have reduced informative value [27]. This can be explained by the fact that predictive methods such as regression, neural nets or classification trees have as a target to produce an equation or a set of rules optimising a fitting criterion. The resulting model is not necessarily intuitive or explanatory, especially when the attributes used for the prediction are highly correlated or when there are violations of the underlying assumptions (e.g. the assumptions of regression). That is why the model sometimes works like a ''black box''. On the other hand, data mining techniques, such as association rules, have as a target to identify correlations and links among data values explaining the interactive behaviour of certain variables.

It is therefore reasonable to investigate whether the combination of explanatory and predictive methods could create an accurate prediction system for software cost estimation. In this study, we explore the combination of two machine learning techniques: Association Rules (AR) and Classification and Regression Trees (CART) in order to benefit from the descriptive and predictive nature of the two methods correspondingly.

AR are used to find logical associations between software development attributes and productivity values describing the influence of various managerial decisions (such as tool selection, development team size), standard project attributes (business type, application type) and product values (size, functionality) to high or low productivity values. CART are used to provide a complete estimation framework and provide an estimate when AR are incapable. The particular methods have been selected because they are governed by the general principles of learning and modelling relationships in data especially in situations were the relationships are complicated, non-linear and sometimes hidden. Their difference from the statistical models is that they describe the associations between variables in terms of explanatory logical relations, than in terms of strict equations.

Potential benefits of the combination of methods involve the ability of the final model to indicate the way particular attributes can increase or decrease productivity in addition with the fact that such assumptions are distinct between different ranges of productivity values. Additionally, the suggested framework provides as a final output a wide set of rules that are reduced according to certain selection criteria and are usually in agreement with the existing intuition of the human user. This approach addresses the problem of typical estimation models which often produce equations or direct estimates that cannot be easily interpreted by the user.

The estimation process involves (a) the definition of a number of ordered productivity categories, (b) the identification of ARs describing the influence of certain project attributes on the software development productivity, (c) the building of a CART model that will be able to classify *all* possible projects to a productivity interval and (d) the transformation of the suggested productivity interval back to a numeric estimate. The suggested approach attempts to take advantage of the unique attributes of the two methods. In particular:

- The method exploits the advantages of AR pattern recognition. Frequent and pertinent relationships among the project attributes and the development productivity are discovered. AR is a method for descriptive modelling and therefore identifies specific relationships often ignored by predictive models such as CART.
- An additional benefit comes from the representation form of AR. Rules are transparent and therefore can be easily understood and rephrased in order to offer a clearer explanation as to how the prediction has been made. This is important to a problem domain such as software cost estimation where the estimator must trust the output otherwise the prediction may be ignored.
- CART method on the other hand as a predictive modelling method, is able to provide a complete estimation framework that based on several historical projects constructs a model that classifies in a reasonable manner *all* projects even the ones with previously unseen attributes.
- CART also avoids overfitting of the model to the historical data information using various ''pruning methods''. CARTs have the benefit of acquiring general knowledge from specific examples of projects.
- The combination of the two methods inherits possible advantages of ML methods. The method provides both numeric and interval estimates of productivity, with a certain probability that the new project belongs to the suggested productivity category.

In order to evaluate the proposed method in terms of prediction accuracy, we make a comparative evaluation of the two classification algorithms alone and combined. Two data sets were used for the evaluation of these approaches. The first one, namely STTF data set, involves maintenance effort data collected from a big commercial bank of Finland. The second data set is ISBSG multi-organizational data set release 7 that contains 1227 projects. The results from the application of the hybrid method

show that both classification and regression accuracy of the method is competitive and even better compared to each method alone.

The paper starts with a presentation of related work in Section 2 and the description of the modelling techniques and the accuracy metrics in Section 3. In Sections 4 and 5, we present the two data sets and we describe in details the process of building and validating the models for the corresponding data sets. In Section 6, we discuss the results and in Section 7, we provide conclusions and directions for future research.

2. Related work

There is a vast literature in the area of software cost estimation. Since 1966 [11], various techniques have been proposed for predicting cost of software development. Surveys on SCE [20] point out that each method has its pros and cons and it can in certain situations be very inaccurate. Expert Judgement (EJ) [17,19,21,31] is the most commonly used technique in software estimation. EJ, is a low cost, easily applied and reliable method based on the validity of the estimator.

Model based techniques such as COCOMO model [5,6] and Function Points (FP) [1,26] are easy to apply but use a formula/function based on several subjective assumptions that cannot be generalized in all situations [24]. Regression models (RM) [14,16,22,34] are often among the most accurate methods but in case of high heteroscedasticity of data present poor performance [32]. Learning oriented techniques such as rule induction decision trees and neural networks have been employed in few studies regarding cost estimation [28,27]. In these studies neural networks are the most accurate method with the main drawback of the method the lack of transparency and the overfitting of the model to the data. AR [4] and CART also applied in the context of a comparative study with Bayesian Networks [38], and Analogy Based Estimation [35,39] in two data sets [3]. In both data sets AR method outperforms the rest of the methods with the disadvantage that the selected rule set could not always provide an estimate. In general, CARTs although able to classify all possible projects usually they do not lead to satisfactory accuracy [22,3] and in a certain study a variant of the method is used to increase estimation accuracy [16].

The fact that none of the above methods was dominant in all situations resulted in further investigation on the conditions under which each method is more accurate and whether the combination of methods is effective. Several studies applied different techniques to different groups of data based on the characteristics of the data set. A particular study [36] concludes that learning techniques perform better when the cost function is discontinuous while model based techniques, in particular stepwise regression, is more effective when the cost function is continuous. Almost similar results are obtained in [32] where it was found that standard multiple regression techniques were best if the data exhibited moderate non-normality but under more extreme conditions such as severe heteroscedasticity, the non-parametric techniques performed best. The comparison of regression, expert judgement and case based reasoning in a homogeneous data set indicated that in cases where one technique predicts poorly, one or both of the others perform significantly better but no mechanism for identifying the situation under which each method performs better was established [29]. From the above studies we can reach to the conclusion that the accuracy of the models is based on the data which are used for their generation.

The combination of multiple estimates produced by either the same or different techniques in order to improve the accuracy of a prediction is also proposed in the literature. Multiple estimates will output a range of possible values that may be combined to produce a more reasonable estimate. The combination of different estimates from different experts is also found in the literature [18,20] indicating that the aggregation of different opinions may be under conditions superior to a single estimate. Issues like the optimum number of experts, their validity (bias) and their low intercorrelation [12] can affect the accuracy of the final aggregated estimate. A procedure for combining such estimates either with mathematical or behavioural models can be found in [9]. In software cost estimation the combination method used is usually a simple average of the estimates [24] or an estimate taking into consideration risk exposure [23].

Combination of multiple techniques in order to create an hybrid method that will provide a single estimate is applied in [25] where cluster analysis with neural networks are integrated. In this study, initially the data were clustered in homogeneous groups. In each one of these groups a neural network was created producing estimates. The results were improved compared to the application of neural networks alone. In [8], the authors suggest a hybrid method, COBRA for software cost estimation. The productivity model has two components: a model that produces a cost overhead estimation and a productivity model based on this cost overhead.

In the present study, AR and CART are combined into one hybrid method, which according to the estimation situation selects one of the two models. Additionally, as the two models provide estimates in intervals, it is possible to combine the two estimates in order to provide a smaller range of estimation values. AR in most cases is the most accurate method but in many other cases cannot provide an estimate. In the later situation, CART model is used to provide an estimate. Finally, we attempt to provide a framework under which each method should be selected.

3. Modelling techniques

3.1. Association rules

Association rules [10] are among the most popular representations of local pattern recognition. They belong to

descriptive modelling and have as a target to describe the data and their underlying relationships with a set of rules that jointly define the target variables [42]. Their target is to find frequent combinations of attribute values that lay in databases. An association rule is a simple probabilistic statement about the co-occurrence of certain events in a database.

Each rule consists of two parts. The left part is the Rule Body (antecedent) and is the necessary condition in order to validate the right part, Rule Head (consequent). Each rule states that if the rule body is true then the rule head is also true with probability p. It is obvious that ARs are Boolean propositions with true or false values. In the rule head, any Boolean expression can be used, but usually conjunction is preferred for simplicity purposes.

Given a set of observations over attributes A_1, A_2,..., A_n in a data set D a simple association rule has the following form :

$$(A_1 = X \wedge A_2 = Y) \Rightarrow A_3 = Z$$

$$\text{confidence} = p(A_3 = Z | A_1 = X,\ A_2 = Y),$$
$$\text{support} = \text{freq}\ (X \cup Y \cup Z, D).$$

This rule is interpreted as following: when the attribute A_1 has the value X and attribute A_2 has the value Y then there is a probability p (confidence) that attribute A_3 has the value Z. For this rule, two major statistics are computed, confidence and support values. Confidence is the probability p defined as the percentage of the records containing X, Y and Z with regard to the overall number of records containing X and Y only. Support is a measure that expresses the frequency of the rule and is the ratio between the number of records that present X, Y and Z to the total number of records in the data set.

AR mining is a two stage process. The first stage involves the identification of all frequent set of attributes contained in the given data set. A set of attributes is frequent if its associated support exceeds a certain support threshold defined by the user. The second stage is generating all pertinent ARs from these itemsets. An AR is pertinent if its associated confidence exceeds a certain confidence threshold specified by the user.

For the estimation of the productivity interval of a project, the rules are sorted according to their confidence values. The first rule whose rule head criteria are satisfied by the attribute values of the project gives the productivity estimate.

3.2. Classification and Regression Trees (CART)

CART is a widely used statistical procedure for producing classification and regression models with a tree-based structure in predictive modelling [7]. The CART model consists of an hierarchy of univariate binary decisions. The algorithm used operates by choosing the best variable for splitting data into two groups at the root node. It can use any one of several different splitting criteria, all producing the effect of partitioning the data at an internal node into two disjoint subsets in such way that the class labels are as homogeneous as possible. This splitting procedure is then applied recursively to the data in each of the child nodes. A greedy local search method to identify good candidate tree structures is used. Finally, a large tree is produced and specific branches of this tree are pruned according to the stopping criteria, so as to avoid overfitting of the data and over-specialization of the model.

CARTs are able to classify not only all projects in the training data set, but unknown projects from a wider group of projects, of which the training projects is presumed to provide a representative example. In our study, a serial algorithm is applied. The decision tree classifier consists of two phases: a *growth* phase and a *prune* phase. In the growth phase, the tree is built by recursively partitioning the data until each partition is either "pure" or sufficient small. The form of the split used to partition the data depends on the type of the attribute used in the split. Only binary splits are considered. Once the tree has been fully grown, it is "pruned" in the second phase to generalize the tree by removing dependence on statistical noise. Pruning a branch of a tree consists of deleting all descendants of the branch except from the root node. The pruning algorithm used is based on the Minimum Description Length principle [33].

In order to estimate the productivity category of a new project, starting from the root node, the right branch is selected according to the project's attributes at each level, moving down until a terminal node is reached.

3.3. Proposed method

The proposed method is a combination of AR and CART method. Fig. 1 depicts the steps of the method. Initially both methods are applied to the training set. AR method exports a certain number of rules exploring patterns involving project attributes unlike CART method which provides an estimation model solely for the dependent variable (productivity or effort). CART model can be used immediately to estimate the productivity or effort value of a new project. In order to provide an estimate exploiting AR patterns a certain procedure has to be followed. The rules with productivity or effort attribute on their rule head and attribute values in their rule body similar to the attribute values of the project under estimation are selected. Also the selected rules have to be frequent and pertinent complying with certain confidence and support thresholds.

The procedure of estimating a new project is simple. Initially the rule set is used in order to provide estimation. If the project attributes are similar to the attributes stated in the rule body then the rule is able to provide estimation. If none of the rules is suitable with the attributes of the new project under estimation then CART estimation model provides a prediction.

AR and CART are both classification methods. Therefore, the dependent variable, is transformed in discrete values. The dependent variable used in this study is for STTF

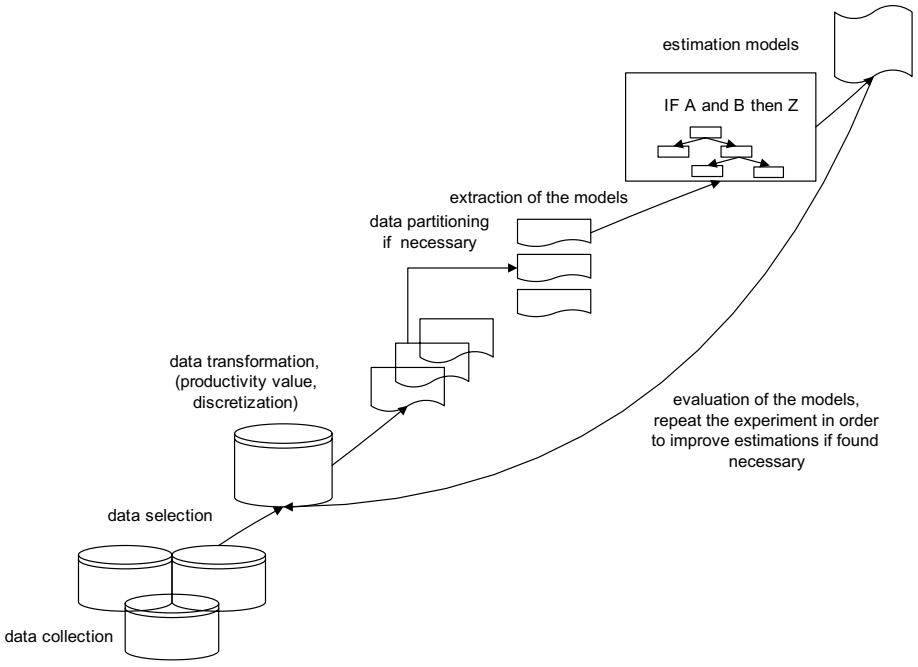

Fig. 1. The various steps of the proposed method.

data set effort value and for ISBSG data set productivity value measured in function points per summary work effort. The problem that arises is which method to use for the discretization of the dependent variable and also how many discrete values should be considered. There is a variety of methods for discretization [41] such as equal width binning, equal frequency binning and clustering. For the number of classes of productivity, Sturge's rule was initially considered: $k = 1 + 3.3\log(n)$ where k is the number of intervals and n is the number of projects in the data set [40].

The above statistical proposals for the width and the number of intervals were validated and configured using many different configurations. The experimental results for STTF data set pointed out that the best configuration is the intervals with equal number of projects while for ISBSG data set the best configuration is the equal intervals of the logarithm of productivity. The number of classes was based on Sturge's rule and were finalized during the generation of the models. When the patterns extracted from the data presented low support and confidence values, intervals of productivity were merged in order to allow the extraction of more statistically powerful assumptions.

Specifically for AR method all variables included in the model have to be discrete. Function points variable was discretized with the same process as mentioned before in both data sets correspondingly. The discretization process of the rest of the variables is mentioned in the next section for each data set separately.

3.4. Accuracy

Three accuracy metrics will be used in order to compare and evaluate the results of the models. In an attempt to evaluate both regression and classification accuracy of the methods we adopt two regression accuracy metrics and one classification accuracy metric. For the regression accuracy of the methods we use the median point of the suggested interval estimate.

The Mean Magnitude Relative Error (MMRE) and the prediction within 25%, PRED(25) will be used for the evaluation of the regression error. The *MMRE* is defined as following: $\frac{1}{n}\sum_{i=1}^{n}\left|\frac{P_i-\hat{P}_i}{P_i}\right|$ where P_i is the actual productivity for project i, $\hat{P}_i$ is the estimate for project i and n is the number of projects. The second regression accuracy metric is *PRED*(25), i.e. the percentage of projects for which the prediction falls within the 25% of the actual value.

For the evaluation of the classification error of the models, *hitrate* will be used [20], i.e. the percentage of projects for which the correct interval has been successfully estimated. Usually the validation of the models is done by selecting the most recent projects as the validation data set and leaving the rest as the training data set. Especially for CART method we also provide the fitting accuracy of the model, which is the hitrate of the model when applied to the training data.

4. Data set 1: STTF data set

4.1. Description of the data

STTF data set comes from a big commercial bank in Finland, which began to collect development and maintenance data as early as 1985. The data were collected by Pekka Forselius and are presented in [30]. Between 1987 and 1995, 250 IBM applications were developed that moved applications from a Bull mainframe environment

to a three-tier architecture system constructed of PCs, local servers and IBM mainframes. From the 250 projects of the database, a subset of 67 applications was presented in [30] that had accurate, complete and valid size, defect and effort data as mentioned in the book.

This data set was selected because it contained projects with many defect data and risk factors whose values have been carefully assessed. The variables of the data set are found in Table 1.

4.2. Preparation of the variables

The preparation of the variables includes the creation of new variables, data modifications, identification of categor-

Table 1
Variable definition in STTF data set

Field name	Full name	Values
correff	Maintenance effort	22–3031
acoreff	Annual maintenance effort	25–3031
totfp	Application size in function points	18–2328
pcobol	Cobol Function Points	0–0.98
ptelon	Telon Function Points	0–0.87
telonuse	Telon is used or not	No, Yes
easyuse	Easy is used or not	No, Yes
peasy	Easytrieve Function Points	0–0.52
pjcl	JCL Function Points	0–0.96
T	Importance of recovery capability (1 = it does not matter, 5 = it is important)	1–4
aggend	Number of maintenance months	8–85
avetrans	Average transactions in 24 h	0–345
avetelev	Level of average transactions in 24 h	1 = Less than 1, 2 = From 1 to 9, 3 = 10 or more
cpu	CPU usage in seconds/24 h	0–2197
cpulev	Level of CPU used	1 = Less than 10, 2 = From 10 to 99 3 = From 100 to 999, 4 = 1000 or more
*r*1	Number of users	Values of risk factors range from 1 to 5
*r*2	Configuration	1 = least risky situation
*r*3	Change management flexibility	5 = most risky situation
*r*4	Structural flexibility	3 does not mean average, it represents a situation
*r*5	Documentation quality	somewhere between least and most risky
*r*6	People dependance	
*r*7	Shutdown time constraints	
*r*8	Online transaction processing integration	
*r*9	Batch processing integration	
*r*10	Capacity flexibility	
disksp	Disk space measured in MB	0–390
dsplev	Level of disk space used	1 = Less than 10, 2 = From 10 to 99 3 = From 100 to 999, 4 = 1000 or more
appdef	Number of errors during 1993	0–163
defects	Classes of defects	0, 1–3, 4–11, >11
borg	Business organization type	BigCorp = Big corporation, Corp = Other corporations Group = Accounting/management, ITServ = IT services InHServ = In-House services, Retail = Retail/people
morg	Internal business unit	Account = Accounting, BUC = Business unit counting Common = Banking service, CustInt = Customer interconnecting Decsup = Decision support, Deposit = Deposit, Payment = Payment TInfra = IT infrastructure, ITServ = IT services ITSupp = IT technical support, IntlBank = International banking LetCred = Letter of credit, Loan = Loan security Person = Personel, Resto = In-house restaurant SecTrade = Securities trading system, Treasury = Treasury
apptype	Application Type	BackOff = BackOffice database Connect = Customer interconnection service Core = Core banking business system InfServ = Information service/decision support
dbms	Database management system	DB2, ISDN
tpms	Transaction Processing management system	BATCH, IIMS, IMS, PTCICS, RECICS

ical variables subsets and the selection of the model to be studied in order to ensure that the results will be comparable with the models presented in [30].

The dependent variable in this analysis is the *acoreff* which corresponds to the corrective maintenance effort during one calendar year. Applications that started maintenance during the year are not strictly comparable with the others so corrective maintenance effort for these applications was adjusted to a full year equivalent. Also variables *avetrans*, *disksp* and *cpu* were transformed into categorical variables based on their levels. Finally, *defect* variable was also transformed to categorical variable as our learning algorithms could not deal with continuous variables. The classes determined contained equal number of projects. Finally, for variables *peasy* and *ptelon* that contained many zero values two new categorical variables were created that signaled if the language was used or not. Definitions for the variables created can be found in Table 1.

4.3. Results

The predictive accuracy of the model was assessed by developing predictive maintenance effort models with data through 1993 and then by testing the models on projects maintained during 1994. The reduced database contained 47 projects at the end of 1993 while the 1994 holdout sample had 19 projects.

The CART model is presented in Fig. 2 and can be explained as following:

- When the number of *function points* is less than 58 then the *effort* is between 25 and 133.5 h.
- For projects with less than 1363.5 fp if they are maintained for less than 49 months the *effort* is between 315.5 and 1057.5 h. Else for projects with less than 1363.5 fp maintained for more than 49 months the *effort* is between 133.5 and 315.5 h.
- In projects with more than 1363.5 fp the *effort* is between 1057.5 and 3031 h.

Function points is the main splitting criterion. Also the maintenance duration (*agend*) seems to affect the corrective maintenance effort. As the maintenance age of a project increases then the annual corrective effort decreases. This can be explained by the fact that most defects, errors, or problems in software appear during the initial usage period of a system. The results coming from the extraction of AR are presented in Table 2.

Five rules have been extracted with support threshold (10.63%, 5 projects) and confidence threshold (70%). Attributes that appear in more than two rules is *r*7, *shut down time constraints* and *r*10, *capacity flexibility*. This set of rules can estimate only 12 out of the 19 projects in the test set. Useful conclusions coming from the rules associate high effort values to projects with increased functionality and high importance of recovery capability. Relatively average effort values are observed to projects that have increased shutdown time constraints but low demands on online transaction processing integration and capacity flexibility. Low effort values are observed when the level of average transactions of the system is low.

In Table 3, we isolate two maintenance projects from the test set that we will attempt to estimate with the help of the models presented in Fig. 2 and Table 2. Initially we try to assess the project maintenance effort using the rule set of Table 2. The first three rules are rejected as their rule body conditions are not satisfied by the projects attributes. The fourth rule is able to estimate the projects effort into the lowest effort interval, between 25 and 133.5 h as the project presents variable *r*10 equal to 1, variable *t* equal to 3 and variable avetelev equal to 1. The examined rule has a confidence value equal to 75 and a support value equal to 12.77 pointing out that the rule classifies correctly 6 out of 8 projects of the data set that present $r10 = 1$, $t = 3$ and $avetelev = 1$ to the effort interval (25, 133.5]. The same effort

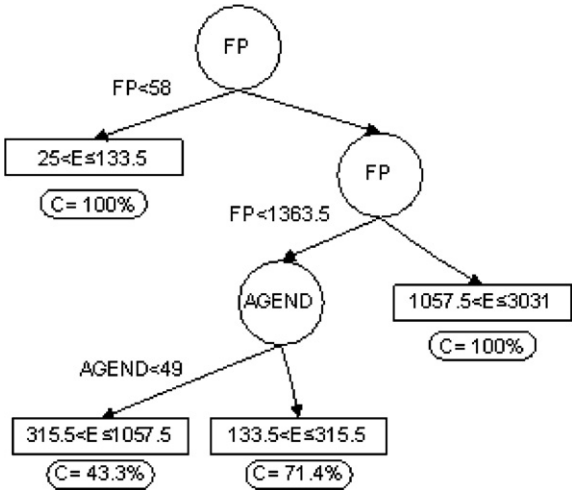

Fig. 2. CART model for STTF data set.

Table 2
Rule set for the estimation of corrective maintenance effort for STTF data set

a/a	Support	Confidence	Rule body	Rule head
1.	10.64	100.00	$T = 4 +$ totfp > 788.5	$1057.5 < E \leqslant 3031$
2.	10.64	83.33	$r7 = 5 + r8 = 1$	$315.5 < E \leqslant 1057.5$
3.	10.64	83.33	$r10 = 1 +$ apptype $=$ Core $+ r7$ in {4, 5}	$133.5 < E \leqslant 315.5$
4.	12.77	75.00	$r10 = 1 + T = 3 +$ avetlev $= 1$	$25 < E \leqslant 133.5$
5.	14.89	70.00	avetlev $= 1 +$ easy $=$ no	$25 < E \leqslant 133.5$

Table 3
Projects under estimation (example)

Id	totfp	Easy	*t*	Age	avetlev	*r*7	*r*8	*r*10	Application type	coreff
16	24	No	3	15	1	5	5	1	Backoff	62
9	448	Yes	4	84	1	4	2	3	InfServ	186

Table 4
Evaluation metrics of the methods for STTF data set

	CART	AR	CART + AR
MMRE	300.48	68.763	59.27
PRED(25)	0.263	0.58	0.473
Hitrate	0.5	0.66	0.894
No. of projects estimated	19	12	19

estimation interval is indicated by CART method as well. While estimating the second project we noticed that none of the rules could provide an estimation. Therefore, in that case we used the CART method to obtain an estimate.

The evaluation results of the hybrid model and the two methods alone are presented in Table 4. The combination of the methods improves the estimation accuracy. It should be mentioned that among the projects in the training data set there was a project (id = 11) for which the actual maintenance effort was very low (9 months). The method placed the project to the lower effort interval but still the variance from the actual value was high, a fact that increased MMRE metric. Had we excluded the particular project from the testet the MMRE would be 30.4.

For comparative reasons we mention the results obtained by regression in [30]. The regression model built to predict maintenance effort involves fp, *r*9, *morg*, *r*3 and *ageend* and has an MMRE of 102.4 and a PRED(25) of 29.4%. Also in [30] the 1993 effort was used to estimate 1994 effort for 17 ongoing maintenance applications (2 applications in the training set started maintenance in 1994 so there were no data for 1993). In that case the MMRE is 75.6 and the PRED(25) 35.3%. The results of the proposed method are improved compared to the results of both models presented in [30].

5. ISBSG data set

5.1. Description of the data

The second data set is ISBSG release 7 [13], a publicly available multi-organizational data set. The International Software Benchmarking Standards Group maintains a repository of international software project metrics to help developers with project estimation and benchmarking. ISBSG data repository release 7 [13] contains 1239 projects that cover the software development industry from 1989 to 2001. The data set contains over 50 fields involving the projects origin, age, context, the type of the product and the project and the development environment the methods and tools utilized.

5.2. Preparation of the variables

The original data set contains 1239 projects. In many records a number of fields are empty or even measured with different approaches. Our target was to include in the study the majority of the projects but also to ensure data validity minimising the variance between the data because of the differences in measurement, or quality, two conflicting targets. The preparation and transformation of data performed involved the selection of projects with data quality rating A and B (projects with data quality rating C were excluded). Projects for which only the development team effort and support was counted and only staff hours were recorded were selected. At this point 556 projects are considered. Partitioning of the data according to their application type was performed. Only the projects with known application type were selected. This further partitioning of data was found necessary due to the variance of data that prevented the extraction of patterns.

At the end of the procedure 288 projects were left and the variables that were involved in the study are presented in Table 5. 46.9% of these projects were Management Information Systems, 17.01% were Transaction/Production Systems, and the rest 36.11% involved various system types, mainly Decision Support Systems, Executive and Office Information Systems, Network Management, Process Control and Real Time applications.

In ISBSG data set, most of the variables that appear in the final training set are discrete, categorical, and therefore suitable for fitting into the methods. The only variable apart form productivity and function points with continuous values is MTS which was discretized empirically. It should be mentioned that variables such as Organization Type, Business Area Type and Application Type presented many different discrete values. For these variables, categories that were found in no more than five records in the whole data set were merged under the label OTHER.

5.3. Results

In this section the extracted models are going to be presented for all approaches and subsets on ISBSG data set.

5.3.1. First data set: Management Information Systems

The first data set contains 135 projects with application type MIS. The model was derived from 128 projects and validated on 7 projects. The model that was derived from the data with CART method is presented in Fig. 3. The splitting nodes are *BAT* and *PPL*. The fitting accuracy of CART to the data is 48.44%.

The rule set derived is presented in Table 6. Nine rules were selected with support and confidence threshold 3.125% (4 projects) and 45.5%, correspondingly. In the rules the most frequent appearing attributes are Business Area Type (*BAT*) and Development Type (*DT*). The number of rules extracted capable to provide estimations for high productivity values was very few, a fact that justifies the low support and confidence values of the particular rules presented in Table 6.

For the extraction of the rules, information coming from CART method was also exploited when grouping together different values of a variable that had the same effect on productivity. As an example of estimation we

Table 5
Variable definition in ISBSG data set

Field name	Full name	Values
FP	Function Points	9–17,518
MTS	Max team size	0–53
DT	Development Type	NewDevelopment, Re-development, Enhancement
LT	Language Type	3GL, 4GL, ApG
PPL	Primary Programming Language	ApG, 4GL, ACCESS, C, C++, CLIPPER, COBOL, CSP, EASYTRIEVE, JAVA, NATURAL, ORACLE OTHER, PERIPHONICS, PL/I, POWERBUILDER, SQL, TELON, VISUALBASIC
OT	Organization Type	Aerospace/Automotive, Banking, Communication, Community Services, Computers, Electricity, Gas, Water, Financial.Business, Government, Insurance, Manufacturing, OTHER, Professional Services, PublicAdministration, Transport&Storage, Wholesale&Retail Trade, Defence, Electronics
DBMS	Database Management System	ACCESS, ADABAS, DB2,IMS, OBJECTSTOR, ORACLE, OTHER
DP	Development Platform	MF, MR, PC
HMA	How Methodology Acquired	Developed/purchased, Developed Inhouse, Purchased
AT	Application Type	DSS, Elect.Data.Interch., Executive.I.S, MIS, Network.M, Office.I.S, OTHER, Process.Control, Real Time, Transaction/Production
BAT	Business Area Type	Accounting, Banking, Engineering, Financial, FineEnforcement, Insurance, Inventory, Legal, Logistics, Manufacturing, OTHER, Personnel, Research&Development, Sales&Marketing, Telecommunications
Implementation Date	Implementation Date	1989–2001

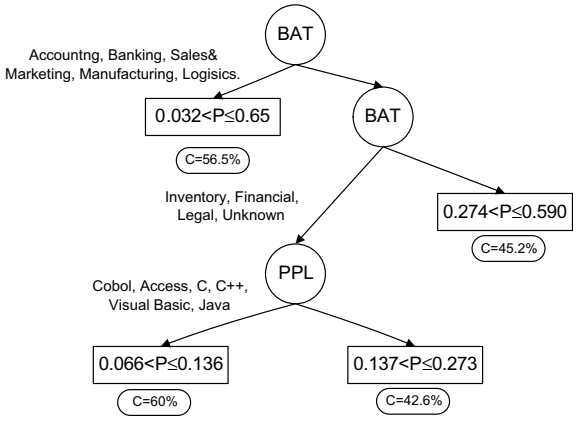

Fig. 3. Productivity estimates with CART for MIS projects.

use project with ID 32583, having the attribute values presented in Table 7.

Taking into consideration the rule, Table 6, the first two rules whose criteria are satisfied are the ones presented in Table 8.

As a final result the method estimates that the productivity is 92.3% likely to be from 0.274 to 5.353 fp/h with a 66.7% probability to be from 0.274 to 0.590 fp/h. CART decision tree also estimates a high productivity value for the particular project as the BAT value of the particular project is not one of the values presented in the left branches of the tree.

5.3.2. Second data set: Transaction Production Systems

The cluster of projects with application type Transaction and Production Systems contained many missing values in the fields of BAT, OT that prevented the extraction of useful patterns. For this subset of projects, we decided to include projects with known BAT and OT values. The total number of the projects in this cluster is 49. The most recent ones starting in 1997 and later were used as the evaluation set. The suggested CART for this situation is presented in Fig. 4.

The fitting accuracy of CART to the data is 55.81. The suggested CART is fairly simple but tends to omit many productivity categories misclassifying the projects belonging to the other productivity intervals. BAT, and DT are the splitting nodes in CARTs. While extracting AR, 8 rules were finally selected. The support and confidence threshold is 4.6% (2 rules) and 50% correspondingly. Frequently appearing attributes in the AR model are MTS, and LT. The rule set is presented in Table 6.

5.3.3. Third data set: other applications

The rest of the projects are 104 involving applications concerning the implementation of Decision Support systems, Office Information systems, Network Management systems, Electronic Data Interchange and Real Time systems. Every discrete category that had less than 5 records in this field was merged under the label "OTHER". The most recent ones, i.e the ones with starting dates 2001 are in the evaluation set.

The structure of the tree produced from CART method is shown in Fig. 5.

The suggested splits are *FP* and *MTS* fields. The rule set extracted is presented in Table 6. The threshold for the support is 3.157% (3 rules) and for confidence is 50.0%.

Table 9 shows the three evaluation metrics for all approaches and subsets on ISBSG data set. As it is obvious in most clusters the method had a hitrate over 70%. In particular for Management Information Systems (first data set) and for the cluster containing the variety of application

Table 6
Association rules for ISBSG data set for all application types

No.	Support	Confidence	Rule body	Rule head
MIS projects				
1.	3.1	100.0	BAT = OTHER and PPL in {APG = 4GL = VB = SQL = TELON = OTHER}	$0.137 < p \leqslant 0.273$
2.	9.4	92.3	PPL = ACCESS	$0.274 < p \leqslant 5.353$
3.	3.1	80.0	DP = MF and $286 <$ FP $\leqslant 629$ and DT in {New = development, Re-development}	$0.066 < p \leqslant 0.136$
4.	3.1	80.0	LT = 4GL and DP = PC and OT = ProfessionalServices and BAT in {Engineering, *p*ersonnel, Research&Development}	$0.274 < p \leqslant 0.590$
5.	8.6	78.5	DBMS = IMS and BAT in {Banking, Accounting, Logistics, Manufacturing, Sales&Marketing}	$0.015 < p \leqslant 0.065$
6.	7.8	76.9	PPL = COBOL and BAT in {Banking, Accounting, Logistics, Manufacturing, Sales&Marketing}	$0.032 < p \leqslant 0.065$
7.	4.0	71.4	LT = 3GL and DBMS = ORACLE	$0.066 < p \leqslant 0.136$
8.	3.1	66.7	BAT = Engineering and PPL in {ACCESS, NATURAL}	$0.274 < p \leqslant 0.590$
9.	2.4	58.3	DT = Enhancement and PPL = SQL	$0.015 < p \leqslant 0.065$
10.	3.9	45.5	LT = 4GL and DBMS = ACCESS	$0.591 < p \leqslant 5.353$
Transaction/Production Systems				
1.	9.3	100.0	$5 \leqslant$ MTS $\leqslant 8$ and $9 \leqslant$ FP $\leqslant 174$	$0.037 < p \leqslant 0.062$
2.	7.0	100.0	LT = 4GL and OT in {Aerospace, ElectricityGasWater Wholesale&Retail Trade, Government, Computer}	$0.231 < p \leqslant 0.321$
3.	7.0	100.0	LT = 4GL and $5 \leqslant$ MTS $\leqslant 8$	$0.137 < p \leqslant 0.273$
4.	4.6	100.0	LT = 4GL and $1 \leqslant$ MTS $\leqslant 4$ and BAT in {Personnel, Insurance, Sales&Marketing, Manufacturing, = Financial, OTHER	$0.231 < p \leqslant 0.783$
5.	18.6	80.0	HMA = Developed/Purchased and PPL in {ACCESS, COBOL, TELON}	$0.037 < p \leqslant 0.062$
6.	20.9	64.3	DT = NewDevelopment and DP = MF and PPL = COBOL	$0.037 < p \leqslant 0.062$
7.	6.8	60.0	DP = MF and LT = 3GL and BAT in {Personnel, Insurance Sales&Marketing, Manufacturing, Financial, OTHER}	$0.169 < \mathrm{p} \leqslant 0.224$
8.	16.3	53.9	OT in {Aerospace, ElectricityGasWater, Wholesale&Retail Trade = Government = Computers}	$0.231 < p \leqslant 0.782$
Various Systems				
1.	4.2	100.0	$114 <$ FP $\leqslant 175$ and MTS = 5-9	$0.051 < p \leqslant 0.096$
2.	5.3	100.0	HMA = DevelopedInhouse and DP = PC and PC = No	$0.438 < p \leqslant 3.102$
3.	4.2	100.0	HMA = Developed = Inhouse and AT = DSS and BAT in {Personnel, Engineering, Financial, Telecommunications}	$0.438 < p \leqslant 3.102$
4.	3.2	100.0	AT = OTHER and DT = New Development and $175 <$ FP $\leqslant 270$	$0.204 < p \leqslant 0.353$
5.	3.2	100.0	MTS = 4 and No and LT = 4GL	$0.204 < p \leqslant 0.353$
6.	3.2	100.0	DP = PC and LT = 4GL and PPL = ACCESS	$0.438 < p \leqslant 0.732$
7.	3.2	75.0	$5 \leqslant$ MTS $\leqslant 9$ and $44 \leqslant$ FP $\leqslant 109$	$0.011 < p \leqslant 0.449$
8.	3.2	60.0	DP = PC and $5 \leqslant$ MTS $\leqslant 9$	$0.103 < p \leqslant 0.191$
9.	3.2	60.0	HMA = DevelopedInhouse and DP = PC and DT = New Development and LT = 4GL	$0.438 < p \leqslant 0.732$
10.	8.4.	50.0	DT = Enhancement and DBMS = ORACLE and AT in {ProcessControl, Executive.I.S}	$0.051 < p \leqslant 0.096$

Table 7
Attribute values for the project under estimation

AT	BAT	DBMS	DT	FP	PPL	LT	MTS	OT	PC	DP
MIS	Engineering	ACCESS	New Development	3	ACCESS	ACCESS	3	Electricity, Gas, water	No	PC

Table 8
Rules that provide estimations for the project

No.	Support	Confidence	Rule head	Rule body
2.	9.4	92.3	PPL_ACCESS	$0.274 < p \leqslant 5.353$
8.	3.1	66.7	BAT_Engineering + PPL in {ACCESS, NATURAL}	$0.274 < p \leqslant 0.590$

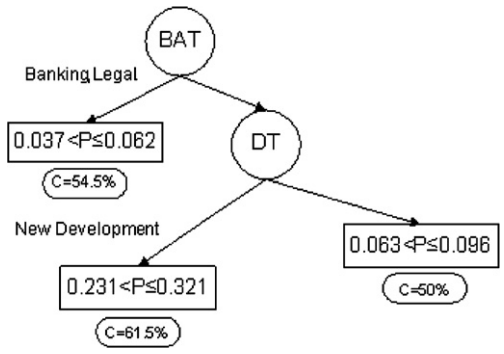

Fig. 4. Productivity estimates with CART for transaction production projects.

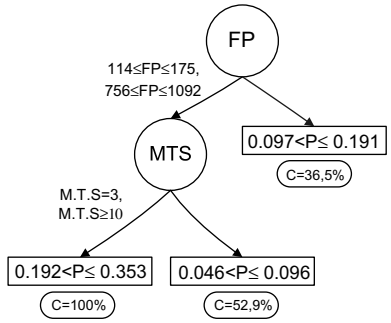

Fig. 5. Productivity estimates with CART for various application types.

types (third data set), the model classifies correctly the majority of the cases. This can be explained by the sufficient number of projects in the training data set and for MIS the fact that the records had relatively low number of missing values in crucial fields such as Business Area Type, Organization type (less than 45% in MIS). Especially for the third cluster although there are many missing values in the above fields (over 72%) and the extracted rules have low values of confidence, the results are satisfactory.

For the data set containing the records involving Transaction Production Systems various difficulties have been encountered. The training set was too small to provide frequently presented patterns having as a consequence the extraction of rules with low support and confidence values and the creation of a relatively week model.

It is useful to present the overall efficiency of the methods in estimating productivity for the totality of the projects of all application types. The results for this case are derived from the overall evaluation metrics for all application types and are presented in Table 9. The combination of methods outperforms each method standing alone. The improvement compared to AR estimation performance is small but quite important for the overall efficiency of the model considering that AR model could provide estimation for only 19 out of 22 projects of the evaluation set.

Although studies on previous releases of ISBSG used point estimates, and as a consequence utilized regression accuracy metrics, it would be useful to present shortly the evaluation results of some of these studies. In [16], which is based on ISBSG release 6 OLS, CART, analogy, ANOVA and robust regression are applied and evaluated with the best method for the estimation of effort being robust regression with MMRE = 0.204 and PRED(25) = 0.67. Another study in the same release is [2] which evaluates categorical regression for effort prediction. Categorical regression has an MMRE equal to 0.40 and PRED(25) = 0.36. In ISBSG release 7 ordinal regression is applied for the estimation of productivity and evaluated in a subset of 52 complete projects. The model has the following results: MMRE = 43.83, PRED(25) = 50% and hitrate = 100%. It should be mentioned that these studies use a very limited set of the projects with complete data, included in ISBSG data set, therefore the results cannot be directly comparable to our study.

6. Discussion

Table 10 indicates the best performing method for each data set for all evaluation metrics. In general, in most cases, the combination of the two methods outperforms each method alone. In the following paragraphs we will attempt to explore the situations under which each of the three methods can be effective.

In general, CART when used alone do not produce satisfactory accuracy results. Only in one case, namely MIS, CART have a comparable accuracy with COMB (i.e. the proposed combination) for PRED(25). This is probably due to the fact that CART produce a tree-like output that typically contains a limited subset of project attributes as nodes. These attributes are the ones that are able to split the data into homogeneous groups referring to the same productivity interval, as much as possible. However, due to this selection mechanism, CART manage to estimate more projects than AR in almost all cases, since CART offer a coarse grain estimation model, into which new projects fit easily.

Table 9
Evaluation metrics for models generated from ISBSG data set

	CART				AR				CART + AR			
	No. of projects estimated	MMRE	PRED(25)	HITRATE	No. of projects estimated	MMRE	PRED(25)	HITRATE	No. of projects estimated	MMRE	PRED(25)	HITRATE
MIS	7/7	29.69	0.71	0.71	5/7	34.26	0.6	0.8	7/7	25.44	0.71	0.71
TRPR	6/6	134.21	0.17	0.17	6/6	33.75	0.33	0.33	6/6	28.64	0.5	0.67
REST	9/9	155.93	0.11	0.22	8/9	21.64	0.75	1	9/9	22.102	0.67	0.89
OVERALL ACCURACY	22/22	106.61	0.33	0.37	19/22	29.88	0.56	0.71	22/22	25.39	0.63	0.76

Table 10
Performance of the estimation models for all metrics in both data sets

	No. of projects estimated	MMRE	PRED(25)	HITRATE
MIS	COMB/CART	COMB	COMB/CART	AR
TRPR	COMB/CART/AR	COMB	COMB	COMB
REST	COMB/CART	AR	AR	AR
STTF	COMB/CART	COMB	AR	COMB
OVERALL ACCURACY	COMB/CART	COMB	COMB	COMB

On the other hand, AR as a descriptive modelling method has the advantage to produce a more refined, multi-rule model that exploits better information contained in project attribute variability. Due to this fact, AR exhibit best accuracy in a specific data set (REST) which is the one that contains the most heterogeneous projects, while they provide good accuracy for specific metrics in other data sets as well (MIS, STTF). Obviously, because of the need to satisfy all conditions of a rule body, it is quite probable that for some projects no rules can be found with adequate support from the data set that will provide an estimate. This means that if the user attempts a priori to create an AR model that always predicts a project, he/she will have to create a very large set of rules including all possible combinations of attribute values. This problem diminishes when the data set used contains many projects and few attributes with limited discrete values. On the other hand, for a particular estimation situation, it is possible to search a rule that will produce an estimate, even with very low support and confidence values. The latter approach is a viable solution, but in many cases it will provide risky estimates and will be very time-consuming.

As already mentioned, the combination of the two methods provides better overall accuracy throughout all data sets examined. The combination of the methods uses initially AR to provide an estimate. As a consequence, all projects that are classified alone by the AR method are also classified to the same interval by the COMB method. Projects that can not be classified by AR are classified by CART. We decided to produce estimates for all projects, although for some of them the probability suggested by CART was low, aiming to produce estimates for all projects in the test data set. However, in a practical estimation situation, the user of the method might choose not to use estimates with low confidence values at all.

In conclusion, we would suggest the use of the combination of the methods for homogeneous data sets, when the models extracted from CART present low confidence values or in the case where few AR rules have been found and with low support values. In our test cases this situation arose in STTF and TR/PR data sets. Intuitively, the combination of the methods is also preferable in the case where one of the above methods estimates a large interval and the other estimates a smaller interval (although we do not provide empirical evidence to support this claim). If a CART or AR model is to be used alone, the branch or the rule to estimate a particular project should have a high confidence value.

Of course, there are certain threats of validity for the study. One of them is the subjective selection of intervals. The study initially was performed with more, relatively shorter intervals. The models extracted in that case presented low confidence values so in many cases the intervals were subjectively concatenated. It is also obvious that the particular characteristics of the data set used for creating the estimation models affect the results and the overall procedure. Another issue is the application to only four data sets, from two cost databases. A wider application could generalize our results and reveal more insight of the methods.

7. Conclusions

In this paper, two methods producing interval estimates have been applied and combined in order to produce a model capable of estimating a productivity interval for a new unknown project. The accuracy of the method has been assessed and compared to the accuracy of the two methods on which it is based, by applying them to four data sets, taken from two cost data bases. Sufficient experience from the use of this novel technique has been acquired, indicating certain advantages of the method.

The proposed method can be appropriate when the models extracted from CART present low confidence values or in the case were AR rules are few with low support values. Also the combination of the methods is suggested in the case of relatively homogeneous data sets.

The hybrid technique is able to exploit the advantages of CART and AR methods and confine some of the disadvantages that each method presents when performing alone. Additionally, a potential advantage of the suggested method is the flexible, user friendly output of the results. The estimate is in the form of rules that the final user can easily understand and modify. For example, a software manager can calibrate the estimate to local environment issues by modifying a rule. Also, he/she can add some new empirical rules based on his/her experience. Therefore, the suggested method can be viewed as a link between expert judgment and purely statistical models, as the model is a result of data analysis that can be easily improved with the combination of expert opinion in the form of rules.

Some directions for future work involve the implementation of statistical methods, like for example the initial partitioning of the projects using cluster analysis in order to apply the CART and AR methodology to subsets containing similar projects. Also, an interesting question is whether and to what extend the method can be improved by using expert judgment as a support. The rules can be derived from automated data analysis and then can be pruned with the help and the expertise of the cost modeller. The expert's role is also to select the intervals into which productivity should be split in the training data, and finally when the estimation is done for a particular project, the expert can select the most probable and representative point-estimates when needed, adjusting the method to the particular data. Finally, we suggest that future studies comparing estimation methods should consider the combination of the AR and CART as a valid alternative estimation method.

Acknowledgement

The authors thank the reviewers for their valuable comments which helped considerably in improving the paper's content, structure and readability.

References

[1] Allan J. Albrecht, John E. Gaffney Jr., Software function, source lines of code, and development effort prediction: a software science validation, IEEE Trans. Softw. Eng. 9 (6) (1983) 639–648.

[2] L. Angelis, I. Stamelos, M. Morisio, Building a software cost estimation model based on categorical data, in: Proceedings of the Eighth IEEE International Symposium on Software Metrics, 2001, pp. 4–15.

[3] S. Bibi, I. Stamelos, L. Angelis, Software cost prediction with predefined interval estimates, in: First Software Measurement European Forum, Rome, Italy, January 2004.

[4] S. Bibi, I. Stamelos, L. Angelis, Software productivity estimation based on association rules, in: Proceedings of the European Software Process Improvement Conference, 13 A.6, Trondheim, Norway, November 2004.

[5] B. Boehm, Software Engineering Economics, Prentice-Hall, Englewood Cliffs, NJ, 1981.

[6] B. Boehm, B. Clark, E. Horowitz, C. Westland, R. Madachy, R. Selby, Cost models for future software life-cycle processes: COCOMO 2.0, in: J.D. Arthur, S.M. Henry (Eds.), Annals of Software Engineering Special Volume on Software Process and Product Measurement, vol. 1, J.C. Baltzer AG, Science Publishers, Amsterdam, The Netherlands, 1995, pp. 45–60.

[7] L. Breiman, J. Friedman, R. Oshlen, C. Stone, Classification and Regression Trees, Wadsworth International Group (1984).

[8] L.C. Briand, K.El. Emam, F. Bomarius, COBRA: a hybrid method for software cost estimation, benchmarking, and risk assessment, in: Proceedings of the 20th International Conference on Software Engineering, Kyoto, Japan, April 19–25, 1998, pp. 390–399.

[9] R. Clemen, R. Winkler, Combining probability distributions form experts in risk analysis, Risk Anal. 19 (2) (1999).

[10] D. Hand, H. Mannila, P. Smyth, Principles of Data Mining, MIT Press, 2001.

[11] O. Helmer-Heidelberg, Social Technology, Basic Books, New York, 1966.

[12] R. Hogarth, A note on aggregating opinions, Organ. Behav. Hum. Perform. 21 (1978) 40–46.

[13] International Software Benchmarking Standards Group, Available from: <http://www.isbsg.org>.

[14] R. Jeffery, I. Wieczorek, A comparative study of cost modeling techniques using public domain multi-organizational and company-specific data, in: Proceedings of ESCOM-SCOPE 2000, Munich, Germany, 18–20 April 2000, Shaker Publishing B.V., pp. 239–248.

[15] R. Jeffery, M. Ruhe, I. Wieczorek, A comparative study of two software development cost modeling techniques using multi-organizational and company-specific data, Inf. Softw. Technol. 42 (14) (2000) 1009–1016.

[16] R. Jeffery, M. Ruhe, I. Wieczorek, Using public domain metrics to estimate software development effort, in: Claes Wohlin (Ed.), Proceedings of the Seventh International Software Metrics Symposium, London, UK, 4–6 April 2001, IEEE Computer Society, Los Alamitos, CA, USA, pp. 16–27.

[17] M. Jørgensen, K.H. Teigen, K.J. Moløkken-Østvold, Better sure than safe? Overconfidence in judgment based software development effort prediction intervals, J. Syst. Softw. 70 (1–2) (2004) 79–93.

[18] M. Jørgensen, K. Moløkken, Combination of software development effort prediction intervals: why, when and how? in: Fourteenth IEEE Conference on Software Engineering and Knowledge Engineering (SEKE'02), Ischia, Italy, 2002.

[19] M. Jorgensen, An effort prediction interval approach based on the empirical distribution of previous estimation accuracy, Inf. Softw. Technol. 45 (2003) 123–126.

[20] Magne Jørgensen, A review of studies on expert estimation of software development effort, J. Syst. Softw. 70 (1–2) (2004) 37–60.

[21] Magne Jorgensen, Practical guidelines for expert-judgment-based software effort estimation, IEEE Software 22 (3) (2005) 57–63.

[22] B. Kitchenham, A procedure for analyzing unbalanced datasets, IEEE Trans. Softw. Eng. 24 (4) (1998) 278–301.

[23] B. Kitchenham, S. Linkman, Estimates, uncertainty and risk, IEEE Software 14 (3) (1997) 69–74.

[24] Barbara Kitchenham, Shari Lawrence Pfleeger, Beth McColl, Suzanne Eagan, An empirical study of maintenance and development estimation accuracy, J. Syst. Softw. 64 (1) (2002) 57–77.

[25] A. Lee, C.H. Cheng, J. Balakrishnan, Software development cost estimation: integrating neural network with cluster analysis, Inf. Manage. 34 (1998) 1–9.

[26] C.J. Lokan, An empirical analysis of function point adjustment factors, Inf. Softw. Technol. (42) (2000) 649–660.
[27] C. Mair, G. Kadoda, M. Lefley, K. Phalp, C. Schofield, M. Shepperd, S. Webster, An investigation of machine learning based prediction systems, J. Syst. Softw. 53 (2000) 23–29.
[28] C. Mair, M. Shepperd, An investigation of rule induction based prediction systems, ICSE'99 Los Angeles, May, 1999.
[29] S. Macdonell, M. Shepperd, Combining techniques to optimize effort predictions in software project management, J Syst. Softw. 66 (2003) 91–98.
[30] K. Maxwell, Applied Statistics for Software Managers, Prentice-Hall, PTR, 2002.
[31] Ursula Passing, Martin Shepperd, An experiment on software project size and effort estimation, International Symposium on Empirical Software Engineering (ISESE'03), 2003, pp. 120–127.
[32] L. Pickard, B. Kitchenham, S.J. Linkmann, Using simulated data sets to compare data analysis techniques used for software cost modeling, IEEE Proc. Softw. 148 (6) (2001) 165–174.
[33] J. Rissanen, Stochastic Complexity in Statistical Inquiry, World Scientific Publ. Co., 1989.
[34] P. Sentas, L. Angelis, I. Stamelos, Software productivity and effort prediction with ordinal regression, J. Inf. Softw. Technol. 47 (1) (2005) 17–29.
[35] M. Shepperd, C. Schofield, B. Kitchenham, Effort estimation using analogy, ICSE (1996) 170–178.
[36] M. Shepperd, F. Kadoda, Comparing software prediction techniques using simulation, TSE 27 (11) (2001) 1014–1022.
[37] K. Srinivisan, D. Fisher, Machine learning approaches to estimating software development effort, IEEE Trans. Softw. Eng. 21 (2) (1995) 126–137.
[38] I. Stamelos, L. Angelis, P. Dimou, E. Sakellaris, On the use of Bayesian belief networks for the prediction of software productivity, Inf. Softw. Technol. 45 (2003) 51–60.
[39] I. Stamelos, L. Angelis, M. Morisio, E. Sakellaris, G. Bleris, Estimating the development cost of custom software, Inf. Manage. 40 (2003) 729–741.
[40] H. Sturge, The choice of class interval, J. Am. Stat. Assoc. (1926) 65–66.
[41] L. Torgo, J. Gama, Regression using classification algorithms, Intell. Data Anal. 1 (4) (1997) 275–292.
[42] I. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, Morgan Kaufmann (1999).