An Evaluation of Technical and Environmental Complexity Factors for Improving Use Case Points Estimation

Ho Le Thi Kim Nhung(✉), Huynh Thai Hoc, and Vo Van Hai

Faculty of Applied Informatics, Tomas Bata University in Zlin,
Nad Stranemi 4511, 76001 Zlin, Czech Republic
{lho, huynh_thai, vo_van}@utb.cz

Abstract. This paper presents a proposed method for improving the prediction ability of the Use Case Points method. Our main goal is to use the Least Absolute Shrinkage and Selection Operator Regression methods to find out which of the technical and environmental complexity factors significantly affect the accuracy of the Use Case Points method. Two regression models were used to calculate the selected significant variables. The results of several evaluation measures show that the proposed estimation method ability is better than the original Use Case Points method. The Sum of Squared Error of the proposed method is better than the results obtained by the original one. The study also enables project managers to understand how to assess the technical and environmental complexity factors better - since they do have an important impact on effort estimation.

AQ1

AQ2

Keywords: Software effort estimation · Use Case Points · Multiple linear regression

1 Introduction

Software Development Effort Estimation is a critical factor in the early phase of the software development life-cycle. Therefore, the success or failure of a software project relates on the accuracy of the estimated effort [1]. The Use Case Points (UCP) method is used for software effort estimation in the early software development stages. Karner was the first to develop the UCP method [9]. The method is based on structured Use Case Diagram descriptions - (scenarios), and analysis of the actors. Structured descriptions -, or transaction-based representation is a prerequisite for UCP. The result obtained by the UCP method in a project is by assigning a "weight" to a specific number of factors that affect it. Then, starting from those factors, one can find out the corresponding effort by software size and fixed productivity factors - (20 person-hours). However, this method has been analysed in literature and it was shown that it has low accuracy in effort estimation [7, 15, 21].

Several researchers have expressed interest in Use Case-based approaches along with their initial applicability. For this reason, many methods have been proposed [6, 8]. These methods are suitable for different complexity, type, domain, environments,

R. Silhavy et al. (Eds.): CoMeSySo 2020, AISC 1294, pp. 1–12, 2020.
https://doi.org/10.1007/978-3-030-63322-6_64

software, etc.; but, they do not always lead to optimal results due to some issues or factors. Most methods only focus on changing the Unadjusted Use Case Point (UUCP) value - which represents the weights assigned to clustered actors and use cases. However, the accuracy of the estimation is affected by many factors.

The UCP method has two adjusting factors - the Technical Complexity Factor (TCF), and the Environmental Complexity Factor (ECF). There are some difficulties when trying to assign values to these adjusting factors. Assigning values to ECF may be difficult since there is often no basis for comparison [16]. Project managers have to guess what was meant by each factor and try to recollect other projects with which this project can be compared. The specific issue is that ECF defines the experience level of each project team. Therefore, it is difficult to suggest that the team evaluate their work - especially when the project managers are estimators and have to assign values to the ECF. Some of the TCF was unclear. For example, factor T10 'Concurrent', showed a certain level of difficulty. This factor could include parallel processing, parallel programming - or whether the system is stand-alone, or interfaces with several other applications. Assigning values to this factor may be not accurate since there are no guidelines in the UCP model to explain exactly what this factor is supposed to measure. The project managers decided that - in this case, it meant interfacing with other systems, and gave the factor a high score. Luis et al. [10], identified the main factors affecting the accuracy of the estimation of the UCP method - which are ECF and TCF. According to this review, these factors affect the estimation accuracy and require a re-evaluation. Therefore, a slight variation in the weight value of the adjusting factors could dramatically affect the software size, and then the estimating effort. Nassif et al. [12] highlighted the need to refine the parameters used as adjusting factors that are directly related to estimations that are calculated using the UCP method.

The rest of this paper is organised as follows: Sect. 2 defines the research questions and evaluation measures. Section 3 presents the methods applied. Section 4 decribes the proposed method. Section 5 shows the experiment evaluation. Section 6 presents the conclusions of this study.

2 Problem Statement

2.1 Research Question and Hypothesis Formulation

The research questions explained by this analysis are as follows:

RQ1: Which correction factors significantly affect the accuracy of the UCP method?
RQ2: Is the proposed method more accurate than the UCP method?

The sum of squares is one of the most important outputs in assessing the model. The general rules is that a smaller sum of squares indicate a better model [18]. Therefore, we assume that the Sum of Squared Errors (SSE) of the proposed Optimising Correction Factors (OCF) method will be significantly lower than the UCP's SSE. A lower SSE indicates that the OCF method can better explain the data-while a higher SSE indicates that the OCF model is poor in explaining the data. In order to

decide whether the method is better capable of estimation, a statistical hypothesis was tested:

$H_o: SSE_{UCP} = SSE_{OCF}$: There is no difference between OCF and UCP estimation capability = No difference in estimation errors.
$H_1: SSE_{UCP} > SSE_{OCF}$: There is a difference between OCF and UCP estimation capability = There is a difference in estimation errors. The SSE of UCP is greater than the SSE of OCF which means the OCF estimation capability outperforms UCP estimation capability.

This paper compares the accuracy of the OCF with that of the UCP, using a pair of two sample t-tests [22]. The paired t-test for two samples is used as a test of the null hypothesis that the means of two normally distributed populations are equal. The t-test will be used for the SSE evaluation.

2.2 Evaluation Criteria

In effort estimation, different criteria are needed to evaluate the effectiveness of models - like standard evaluation. The accuracy in terms of the Mean of Magnitude of Relative Error (MMRE) and Percentage of Prediction within x% (PRED(x)) are the two most common metrics proposed and used in software engineering [6]. Both parameters are based on the so-called Magnitude of Relative Error (MRE) quantity. The Sum of Squares Errors (SSE) - is an important metric for measuring modeling error variations [18]. The equations are given as follows:

$$MRE = \frac{|\hat{y}_i - y_i|}{y_i} \tag{1}$$

$$MMRE = \frac{\sum_i^N MRE}{N} \tag{2}$$

$$PRED(0.25) = \frac{1}{N}\sum_i^N \begin{cases} 1, & if \quad MRE_i < 0.25 \\ 0, & otherwise \end{cases} \tag{3}$$

$$SSE = \sum_i^N \varepsilon_i^2 \tag{4}$$

Where, N is the number of observations, y_i is the known real value, $\hat{y}_i$ is the predicted (estimated) value, $\bar{y}$ is the mean value of predicted values, and ε is the residual error value. The purpose of these results is to keep MMRE, and SSE minimalised and PRED (0.25) maximised. The most important criterion for the scope of our evaluation is the SSE minimal value, because it means that the OCF method attained optimal performance.

3 Methods

3.1 Use Case Points

The UCP is calculated by computing four basic size metrics [9]. The first metric is obtained from the Unadjusted Actor Weight (UAW); shown in Eq. (5).

$$\text{UAW} = \sum_{i=1}^{3} a_i \times w_i \tag{5}$$

Where, a_i is the number of actors in the i^{th} actor type and w_i is the associate complexity weight for each type. The actors are classified into three levels based on complexity in Table 1.

The unadjusted use case weight (UUCW) is calculated as shown in Eq. (6).

$$\text{UUCW} = \sum_{j=1}^{3} uc_j \times w_j \tag{6}$$

Where, uc_j is the number of use case in the j^{th} use case type and w_j is the associate complexity weight for each type. The use case complexity is classified into 3 categories, based on the number of transactions in the use case, (see Table 2).

The Technical Complexity Factor (TCF) in Eq. (7) can be calculated from the set of 13 factors.

$$\text{TCF} = 0.6 + \left(0.01 \times \sum_{i=1}^{13} t_i \times fw_i\right) \tag{7}$$

Where, t_i is the value of complexity factor i, and fw_i is the weight of factor i. TCF is considerd as a correction value that describes a set of important factors for the project. Table 3 presents the TCF as defined in the UCP.

The second correction value is based on the environmental complexity factor (ECF) in Eq. (8).

$$\text{ECF} = 1.4 - \left(0.03 \times \sum_{i=1}^{8} e_i \times ew_i\right) \tag{8}$$

Where, e_i is the value of complexity factor i, and ew_i is the weight of factor i. ECF is computed from a set of 8 factors that describe the non-functional requirements. Table 4 depicts the environmental factors.

The final result of the estimation (UCP) in Eq. (9) is calculated as an aggregate of four metrics UAW, UUCW, TCF and ECF.

$$\text{UCP} = (\text{UAW} + \text{UUCW}) \times TCF \times ECF \tag{9}$$

Table 1. Actor classification and their weights.

Actor classification	Complexity weight
Simple	1
Average	2
Complex	3

Table 2. Use case classification and their weights.

Use case classification	Number of transactions	Complexity weight
Simple	(0–4)	5
Average	<4–7>	10
Complex	(7–∞)	15

Table 3. Technical complexity factors

Factor ID	Description	Weight
T1	Distributed system	2
T2	Response adjectives	2
T3	End-user efficiency	1
T4	Complex processing	1
T5	Reusable code	1
T6	Easy to install	0.5
T7	Ease of use	0.5
T8	Portable	2
T9	Easy to change	1
T10	Concurrent	1
T11	Security feature	1
T12	Access for third parties	1
T13	Special training required	1

Table 4. Environmental complexity factors

Factor ID	Description	Weight
E1	Familiar with RUP	1.5
E2	Application experience	0.5
E3	Object-oriented experience	1
E4	Lead analyst capability	0.5
E5	Motivation	1
E6	Stable requirements	2
E7	Part-time workers	−1
E8	Difficult programming language	2

3.2 Regression Shrinkage and Selection Using LASSO

This research paper uses the Least Absolute Shrinkage and Selection Operator - (LASSO) to address variable selection in regression analysis [4, 13, 14]. The technical and environmental complexity factors - (considered as correction values) that describe a set of important factors for the project in the UCP method are analysed. We evaluate these correction values in order to determine their contribution to effort estimation. The correlations to real project size and correction value correlations are measured.

LASSO regression is a method that performs two main tasks: L1 - regularisation and feature selection. It forms a constraint on the sum of the absolute values of the model variables, where the sum is required less than an upper bound (a fixed value). The method aplies a shrinking - (regularisation) process which penalises regression variables - (correction value) coefficients by shrinking some of them to zero.

During the feature selection process, the correction values that still have a non-zero coefficient after the shrinking process are selected to form part of the model. The goal of this process is to minimise prediction error - (the sum of squared errors - with an upper bound on the sum of the absolute values of the model parameters). The LASSO method is defined by the solution to the l_1 optimisation problem - (the formulation used by Buhlmann et al. [5]):

$$minimize\left(\frac{\|Y - X\beta\|_2^2}{n}\right) \qquad \text{subject to } \sum_{j=1}^{k} \|\beta\|_1 < t \tag{10}$$

Where, t is the upper bound for the sum of the coefficients. This optimisation problem is equivalent to the parameter estimation below:

$$\hat{\beta}(\lambda) = argmin_{\beta}\left(\frac{\|Y - X\beta\|_2^2}{n} + \lambda\|\beta\|_1\right) \tag{11}$$

Where, $\|Y - X\beta\|_2^2 = \sum_{i=0}^{n}\left(Y_i - (X\beta)_i\right)^2$, $\|\beta\|_1 = \sum_{j=1}^{k}\left|\beta_j\right|$ and $\lambda \geq 0$ is the variable that controls the strength of the penalty; the larger the value of λ, the greater the amount of shrinkage. The relation between λ value and fixed value t is a reverse relationship. It is certain that, as t becomes infinity, the problem becomes an ordinary least squares; and λ will become 0. However, this is vice versa when t reaches 0, all coefficients will shrink to 0; and λ will go to infinity.

In this research paper, we use LASSO regression for its variable selection properties. When we minimise the optimisation problem, some coefficients are shrunk to zero - i.e. $\hat{\beta}_i(\lambda) = 0$; for some values of j - (depending on the value of parameter λ). In this way, features with coefficients equal to zero are excluded from the model.

4 Optimising Correction Factors

This section describes how to optimise the UCP method by considering correction factors. The proposed method improves the accuracy of the estimation by suggesting a new formula for calculating the correction factors in the UCP method. We have called this the Optimising Correction Factors (OCF) approach.

OCF can be divided into three phases. The LASSO-based Selection Phase - (Phase I), applies the LASSO regression with the determined regularisation parameter λ to extract a selected variable set; as shown in Eq. (11). In practice, the tuning parameter λ, which controls the strength of the penalty, assumes great importance. Indeed, when λ is sufficiently large then coefficients are forced to be exactly equal to zero, and so dimensionality can be reduced in this way. The larger the parameter λ is, the more the number of coefficients is shrunk to zero. In this phase, we determined the appropriate λ value using the Leave One Out Cross-Validation (LOOCV) technique [19, 20]; where the R-Square attains its maximum value. This technique is used because of its deterministic property - and suitability for small datasets.

LASSO regression is used to obtain the TCF and ECF correction coefficients - as described in Eq. (12) and Eq. (13) respectively.

$$y\_TCF_i = \alpha_0 + \sum_{i=1}^{13} \alpha_i \times t_i \times fw_i \tag{12}$$

$$y\_ECF_i = \beta_0 + \sum_{i=1}^{8} \beta_i \times \mathrm{e}_i \times ew_i \tag{13}$$

Where, y_TCF_i be $\left(\frac{Real\_P20}{(UAW+UUCW)\times ECF} - 0.6\right) \times \frac{1}{0.01}$, y_ECF_i be $\left(\frac{Real\_P20}{(UAW+UUCW)\times TCF} - 1.4\right) \times \frac{-1}{0.03}$, and $\alpha_0, \alpha_i\ \beta_0, \beta_i$ are the regression coefficient parameters obtained from the LASSO regression; $Real\_P20$ is the real size of software projects from historical datasets. The LASSO-based selected variables in TCF and ECF are designated as LaTF and LaEF respectively - (see Table 5 and Table 6).

The Model Fitting Phase - (Phase II), here, the regression model has the selected variable sets (LaTF and LaEF). Least Squares Regression (LSR) is used to obtain the coefficients for LaTF and LaEF in Eq. (14) and Eq. (15) respectively. LaTF and LaEF values represent the final technical and environmental complexity factors - (correction factors), in the OCF method.

$$y\_LaTF_i = \alpha_0 + \sum_{i=1}^{n} \alpha_i \times LaT_i \times WLt_i \tag{14}$$

$$y\_LaEF_i = \beta_0 + \sum_{i=1}^{m} \beta_i \times \mathrm{LaE}_i \times WLe_i \tag{15}$$

Where, let y_LaTF_i and y_LaEF_i be the TCF and ECF from the historical dataset; n is the number of LaTF; m is the number of LaEF; and $\alpha_0, \alpha_i\ \beta_0, \beta_i$ are regression coefficient parameters obtained from LSR.

LaTF and LaEF are obtained according to Eq. (16) and Eq. (17).

$$\text{LaTF} = \alpha_0 + \sum_{i=1}^{n} \alpha_i \times LaT_i \times WLt_i \tag{16}$$

$$\text{LaEF} = \beta_0 + \sum_{i=1}^{m} \beta_i \times \text{La}E_i \times WLe_i \tag{17}$$

The Use Case Point Estimation Phase - (Phase III) is determined. The effort estimation final result of the proposed OCF method is described by Eq. (18). This is calculated as the aggregate of four UAW metrics - (viz Eq. (5), UUCW (viz Eq. (6), LaTF (viz Eq. (16), and LaEF (viz Eq. (17).

$$UCP_{OCF} = (\text{UAW} + \text{UUCW}) \times LaTF \times LaEF \tag{18}$$

Table 5. LASSO-based technical factors (LaTF)

LaT_i	Description	Weighting factor (WLt)
LaT1	Distributed system	2
LaT2	End-user efficiency	1
LaT3	Complex processing	1
LaT4	Reusable code	1
LaT5	Easy to install	0.5
LaT6	Ease of use	0.5
LaT7	Easy to change	1
LaT8	Concurrent	1
LaT9	Security feature	1

Table 6. LASSO-based environmental factors (LaEF)

LaE_i	Description	Weighting factor (WLe)
LaE1	Application experience	0.5
LaE2	Object-oriented experience	1
LaE3	Lead analyst capability	0.5
LaE4	Motivation	1
LaE5	Stable requirements	2
LaE6	Part-time workers	−1
LaE7	Difficult programming language	2

5 Experiment Evaluation

5.1 Project Dataset

The OCF method is evaluated according to the dataset prepared by Silhavy et al. [17], which contains data on 70 projects. We also use a standard of 20 person-hours per UCP [9] for comparison and estimation purposes, as an efficiency comparison of the methods, without considering the productivity factor. These dataset characteristics are described in Table 7.

Table 7. Dataset characteristics

	Median man-hours	Median Real_P20	Standard deviation	Minimum Real_P20	Maximum Real_P20	n
Dataset	6406	320.3	33.21	288.75	398.5	70

5.2 Model Evaluations

Goodness of fit was evaluated with an adjusted R^2. Table 8 shows fitted models with the selected variables.The LASSO-based selection dropped four factors - (T2, T8, T12, T13) in TCF; and one factor - (E2) in ECF.

The remained variables were statistically significant at $\lambda_{TCF} = 0.001223$ and $\lambda_{ECF} = 0.002413$. The adjusted $R^2 = 0.89$ also describes that the goodness of fit for LaTF and LaEF regression models that contain selected variables. Table 9 shows the regression formulas obtained for each model. LaTF and LaEF models are linear models.

Table 8. Variable selection results in TCF and ECF

Variable selection in TCF		Variable selection in ECF	
(Intercept)	8.108256	(Intercept)	0.73294
T1	0.958168	E1	–
T2	–	E2	0.281381
T3	1.051899	E3	1.108032
T4	0.964772	E4	1.097029
T5	0.998221	E5	0.965594
T6	0.995997	E6	0.946942
T7	1.006057	E7	0.943941
T8	–	E8	0.962185
T9	0.996075	Adjusted R^2	0.89
T10	1.001304		
T11	0.979878		
T12	–		
T13	–		
Adjusted R^2	0.89		

Table 9. Regression formulae: LaTF and LaEF models

$LaTF \sim 0.05674 + 0.00953 * LaT_1 * WFt_1 + 0.010582 * LaT_2 * WFt_2 + 0.009666 * LaT_3 * WFt_3 + 0.009928 * LaT_4 * WFt_4 + 0.011672 * LaT_5 * WFt_5 + 0.001985 * LaT_6 * WFt_6 + 0.020019 * LaT_7 * WFt_7 - 0.0303 * LaT_8 * WFt_8 + 0.168137 * LaT_9 * WFt_9$
$LaEF \sim 1.380776 - 0.010579 * LaE_1 * WFe_1 - 0.033354 * LaE_2 * WFe_2 - 0.032892 * LaE_3 * WFe_3 - 0.029148 * LaE_4 * WFe_4 - 0.028628 * LaE_5 * WFe_5 - 0.028577 * LaE_6 * WFe_6 - 0.029007 * LaE_7 * WFe_7$

5.3 The OCF and UCP Method Comparison

In this section, OCF is compared with UCP - where UCP represents Karner's UCP method - (viz Eq. (9)). As can be seen in Table 10, the OCF method is demonstrably better than the UCP method with MMRE, PRED(0.25) and SSE. The PRED(0.25) OCF method is more than 60% better than the result obtained by the UCP method. The SSE and MMRE of OCF are both better than UCP. Therefore, one can conclude that the OCF method can produce more accurate estimations.

Table 10. Comparison of estimation method performances.

	UCP	OCF
MMRE	0.527	0.283
PRED(0.25)	0.38	0.66
SSE	236881.001	202202.185
n	70	70

The t-test result of the hypothesis is given in Table 11. The hypothesis computational test results - (MRE, MMRE, PRED and SSE), show that OFC is the best in terms of estimation accuracy. SSE was tested at the 5% significance level, and the p-value - (0.0245), in this scenario is below the 5% significance level. This means that one can reject the null hypothesis. Thus, leading one to accept the alternative hypothesis (H1), which states that the OCF method is more accurate than the UCP method.

Table 11. t-test hypothesis results.

Degree of freedom	t Value	p Value
69	2.0037	0.0245

6 Conclusion

The effect of the TCF and ECF correction values were studied, and the OCF method for software development effort estimation was proposed.

As for RQ1, it can be shown that LaTF and LaEF impacted estimation accuracy. The significantly contributed factors in LaTF are the Distributed System - (LaT1), End-user Efficiency (LaT2), Complex Processing (LaT3), Reusable Code (LaT4), Installation Ease (LaT5), Ease of Use (LaT6), Portablity (LaT7), Concurrence (LaT8), and Security Feature (LaT9). The significant contributive factors in LaEF are Application Experience (LaE1), Object-Oriented Experience (LaE2), Lead Analyst Capability (LaE3), Motivation (LaE4), Stable Requirements (LaE5), Part-time Workers (LaE6), and Difficult Programming Language (LaE7). The results are valid for the experimental dataset described herein.

Both LaTF and LaEF are significant for estimation purposes since they affect size estimation and allow the OCF method to increase accuracy as compared to the UCP method.

As regards RQ2, it can be concluded that the OCF method is better than the UCP method. The OCF method estimation accuracy is outperformed - (viz Table 10). When comparing OCF SSE to the OCF method, one can see that this provides a more than 15% improvement. Similarly, the UCP method had a PRED(0.25) of 0.38; while the OCF method achieved a PRED(0.25) of 0.66.

More datasets will be tested in future work. Comparisons with Machine Learning methods will also be performed.

Acknowledgment. This work was supported by the Faculty of Applied Informatics, Tomas Bata University in Zlín, under Project SV13202001020-PU30, Project IGA/CebiaTech/2020/001, and Project RVO/FAI/2020/002.

References

1. Boehm, B.W.: Software Engineering Economics. Prentice-Hall, Englewood Cliffs (1981)
2. Kutner, M.H., Nachtsheim, C.J., Neter, J.: Applied Linear Regression Models, 4th edn. McGrawHill Companies, Inc., New York (2004) AQ3
3. Sermpinis, G., Serafeim, T., Ping, Z.: Modelling market implied ratings using LASSO variable selection techniques. J. Empir. Finance **48**, 19–35 (2018)
4. Tibshirani, R.: Regression shrinkage and selection via the LASSO. J. Roy. Stat. Soc. Ser. B **58**(1), 267–288 (1996)
5. Bühlmann, P., Van de Geer, S.: Statistics for High-Dimensional Data: Methods. Theory and Applications. Springer, Heidelberg (2011)
6. Le Thi Kim Nhung, H., Hoc, H.T., Van Hai, V.: A review of use case-based development effort estimation methods in the system development context. In: Silhavy, R., Silhavy, P., Prokopova, Z. (eds.) Computational Statistics and Mathematical Modeling Methods in Intelligent Systems. CoMeSySo 2019. Advances in Intelligent Systems and Computing, vol. 1047. Springer, Cham (2019)
7. Silhavy, R., Silhavy, P., Prokopova, Z.: Algorithmic optimisation method for improving use case points estimation. PLoS ONE **10**, e0141887 (2015)

8. Saroha, M., Sahu, S.: Analysis of various software effort estimation techniques. Int. Res. J. Comput. Electron. Eng. (IRJCEE) **3**(2), 1–7 (2015)
9. Karner, G.: Resource estimation for objector projects. Objective Systems (1993)
10. Luis, M.H., Sussy, B.O.: Factors affecting the accuracy of use case points. In: Trends and Applications in Software Engineering. Advances in Intelligent Systems and Computing, vol. 537 (2017)
11. Singh, J., Sahoo, B.: UML based object-oriented software development effort estimation using ANN. National Institute of Technology Rourkela, Rourkela (2012)
12. Nassif, A.B., Ho, D., Capretz, L.F.: Towards an early software estimation using log-linear regression and a multilayer perceptron model. J. Syst. Softw. **86**(1), 144–160 (2013)
13. Valeria, F.: Feature selection using LASSSO. Research paper in Business Analytics (2017)
14. Muthukrishnan, R., Rohini, R.: LASSO: a feature selection technique in predictive modeling for machine learning. In: IEEE International Conference on Advances in Computer Applications (ICACA) (2016)
15. Hoc, H.T., Van Hai, V., Le Thi Kim Nhung, H.: A review of the regression models applicable to software project effort estimation. In: Silhavy, R., Silhavy, P., Prokopova, Z. (eds.) Computational Statistics and Mathematical Modeling Methods in Intelligent Systems. CoMeSySo 2019. Advances in Intelligent Systems and Computing, vol. 1047. Springer, Cham (2019)
16. Bente, A., Hege, D., Dag, I.K.S., Magne, J.: Estimating software development effort based on use cases experiences from industry. In: 4th International Conference on the Unified Modeling Language, Modeling Languages, Concepts, and Tools, Toronto, Canada, 1–5 October 2001 (2001)
17. Silhavy, R., Silhavy, P., Prokopova, Z.: Analysis and selection of a regression model for the use case points method using a stepwise approach. J. Syst. Softw. **125**, 1–14 (2017)
18. Golberg, M.A., Cho, H.A.: Introduction to Regression Analysis. Wessex Institute of Technology, WIT Press (2010)
19. Van der Laan, M.J., Dudoit, S.: Unified cross-validation methodology for selection among estimators and a general cross-validated adaptive epsilon-net estimator: finite sample oracle inequalities and examples. U.C. Berkeley Division of Biostatistics Working Paper Series (2003)
20. Van der Vaart, A.W., Dudoit, S., Van der Laan, M.J.: Oracle inequalities for multi-fold cross-validation. To appear in Statistics and Decisions (2006)
21. Van Hai, V., Le Thi Kim Nhung, H., Hoc, H.T.: A review of software effort estimation by using functional points analysis. In: Silhavy, R., Silhavy, P., Prokopova, Z. (eds.) Computational Statistics and Mathematical Modeling Methods in Intelligent Systems. CoMeSySo 2019. Advances in Intelligent Systems and Computing, vol. 1047. Springer, Cham (2019)
22. Ross, A., Willson, V.L.: Paired Samples T-Test in Basic and Advanced Statistical Tests. SensePublishers, Rotterdam (2017)

Author Query Form

Book ID : **490019_1_En**

Chapter No : **64**

Please ensure you fill out your response to the queries raised below and return this form along with your corrections.

Dear Author,
During the process of typesetting your chapter, the following queries have arisen. Please check your typeset proof carefully against the queries listed below and mark the necessary changes either directly on the proof/online grid or in the 'Author's response' area provided below

Query Refs.	Details Required	Author's Response
AQ1	This is to inform you that corresponding author has been identified as per the information available in the Copyright form.	
AQ2	Kindly check and confirm the authors given names and family names are correctly identified in author group. Amend if necessary.	
AQ3	References [2, 3, 11] are given in the list but not cited in the text. Please cite this in text or delete this from the list.	

MARKED PROOF

Please correct and return this set

Please use the proof correction marks shown below for all alterations and corrections. If you wish to return your proof by fax you should ensure that all amendments are written clearly in dark ink and are made well within the page margins.

Instruction to printer	*Textual mark*	*Marginal mark*
Leave unchanged	··· under matter to remain	New matter followed by
Insert in text the matter indicated in the margin		or (x2)
Delete	through single character, rule or underline or through all characters to be deleted	or (x2)
Substitute character or substitute part of one or more word(s)	through letter or through characters	new character or new characters
Change to italics	under matter to be changed	
Change to capitals	under matter to be changed	
Change to small capitals	under matter to be changed	
Change to bold type	under matter to be changed	
Change to bold italic	under matter to be changed	
Change to lower case	Encircle matter to be changed	
Change italic to upright type	(As above)	
Change bold to non-bold type	(As above)	
Insert 'superior' character	through character or where required	or under character e.g. or
Insert 'inferior' character	(As above)	over character e.g.
Insert full stop	(As above)	
Insert comma	(As above)	
Insert single quotation marks	(As above)	or and/or or
Insert double quotation marks	(As above)	or and/or or
Insert hyphen	(As above)	
Start new paragraph		
No new paragraph		
Transpose		
Close up	linking characters	
Insert or substitute space between characters or words	through character or where required	
Reduce space between characters or words	between characters or words affected	