Information and Software Technology 97 (2018) 1–9

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

Evaluating subset selection methods for use case points estimation

Radek Silhavy*, Petr Silhavy, Zdenka Prokopova

Tomas Bata University in Zlin, Faculty of Applied Infomatics, Nad Stranemi 4511, Zlin 76001, Czech Republic

A R T I C L E I N F O

Keywords:
Software Development Effort Estimation
Software size estimation
Clustering techniques
Spectral Clustering
K-means
Moving Window
Use Case Points

A B S T R A C T

When the Use Case Points method is used for software effort estimation, users are faced with low model accuracy which impacts on its practical application. This study investigates the significance of using subset selection methods for the prediction accuracy of Multiple Linear Regression models, obtained by the stepwise approach. K-means, Spectral Clustering, the Gaussian Mixture Model and Moving Window are evaluated as appropriate subset selection techniques. The methods were evaluated according to several evaluation criteria and then statistically tested. Evaluation was performing on two independent datasets - which differ in project types and size. Both were cut by the hold-out method. If clustering were used, the training sets were clustered into 3 classes; and, for each of class, an independent regression model was created. These were later used for the prediction of testing sets. If Moving Window was used, then window of sizes 5, 10 and 15 were tested.

The results show that clustering techniques decrease prediction errors significantly when compared to Use Case Points or moving windows methods. Spectral Clustering was selected as the best-performing solution, because it achieves a Sum of Squared Errors reduction of 32% for the first dataset, and 98% for the second dataset. The Mean Absolute Percentage Error is less than 1% for the second dataset for Spectral Clustering; 9% for moving window; and 27% for Use Case Points. When the first dataset is used, then prediction errors are significantly higher – 53% for Spectral Clustering, but Use Case Points produces a 165% result.

It can be concluded that this study proves subset selection techniques as a significant method for improving the prediction ability of linear regression models - which are used for software development effort prediction. It can also be concluded that the clustering method performs better than the moving window method.

1. Introduction

Software Development Effort Estimation methods represent actual topics that are crucial for software project planning. If effort estimation is not performed, it causes incorrect project planning, which is an important risk in software engineering project management.

At the early stage, only limited knowledge about a software project is available. The project scope is described in a limited way, and usually, there is no possibility of obtaining a complex and valid information - which should be useful for estimation purposes. Therefore, the Use Case Point (UCP) method [1], can be used to perform estimations. UCP is based on Use Case Model (UCM) structured scenario and actors analysis. Structured scenarios or transactions based representation is a prerequisite for UCP. The actors and use cases are evaluated; the software size is estimated based on an evaluation of the UCM. UCP considers software development efforts as linear correlated value to software size [2].

In previously published studies authors presents improvements [2–5] and simplification [6–8] of an estimation process.

In this paper, the focus is on the issue of "how can we automatically come up with the optimum subset of historical data-points?" Methods for finding similarities or analogies between historical data-points can lead to reducing an estimation error.

The rest of the article is structured as follows: Section 1 is an Introduction. Section 2 defines the research questions. Section 3 describes the methods proposed to address the research questions. Section 4 introduces the project datasets and experimental procedure. Section 5 presents the results obtained. Finally, Section 6 summarises the conclusions and future work.

1.1. Related work

Jorgensen and Shepperd in [9], identify 11 estimation approaches, which are mostly based on statistical or numerical models. Many of them are based on linear regression. Many these algorithms are based on historical datasets, where all available projects are considered for new estimation. Silhavy et al. [2], present the Algorithmic Optimisation Method (AOM), which is based on linear regression and brings

* Corresponding author.
E-mail addresses: radek@silhavy.cz, rsilhavy@fai.utb.cz (R. Silhavy).

https://doi.org/10.1016/j.infsof.2017.12.009
Received 30 June 2017; Received in revised form 18 December 2017; Accepted 21 December 2017
Available online 28 December 2017

significant improvements in comparison to UCP. The AOM methods outperform many other methods because of its simplicity; only the variables known from UCP are used in AOM. The second group represents Machine Learning Algorithms. Wen et al. [10], conducted a systematic literature review in which they identify 84 primary studies and 8 methods of applied machine learning algorithms. In this paper, the focus is on methods which are applicable in conjunction with the AOM method – or, more generally, with Least Squared Regression models (LSR), or Multiple Linear Regression (MLR).

In [11], Silhavy et al., recently discussed variable validity and model selection. They declare that all variables from the UCP method are understood as exploratory variables, and the stepwise procedure allows one to select the best performing model.

Idri et al. [12], provide a systematic mapping study of Analogy-based Software Development Effort Estimation (ASEE). This paper´s authors investigate 65 studies from 1990 to 2012. Most of these studies are aimed at subset method selection. Idri et al., conclude that ASEE outperforms most algorithms which used all historical data-points available. ASEE method looks for similarities in historical projects. Clustering helps to find analogy among projects - and is a broadly investigated method for reducing the number of historical data-points and selecting the most similar subset. Azzeh and Nassif [13], deal with setting the number of nearest projects. These authors recommend a method called Bisecting k-medoids Clustering and have claimed that this method is better than common ASEE methods.

Azzeh et al. in their paper [14], present a hybrid model that consists of classification and prediction stages using a Support Vector Machine and Radial Basis Neural Networks. They compare the said model with k-medoids. They recommend that ECF be omitted from the estimation and to focus all estimation on the productivity factor - which represents the ratio between UCP, and development effort in person-hours.

Bardisiri et al. [15], declare that clustering has a significant effect on the accuracy of development effort estimation because it allows one to omit irrelevant projects from historical data-points.

Prokopova et al. [16], compare k-means, hierarchical and density-based clustering techniques with three different distance metrics. The results show that all tested clustering techniques improve estimation accuracy and that the number of clusters plays a significant role. It is important to select the clustering type and distance metric properly. The authors show that hierarchical clustering has produced inappropriate distribution of clusters – and therefore, cannot be used. The k-means clustering technique with 4 clusters and the cosine distance metric appears to be the most appropriate method for a tested dataset and AOM model.

In [17], Bardisiri et al., improved their method by the combination of ASEE and the Particle Swarm Optimization (PSO) [18] algorithm. They introduce a weighting system in which the project attributes of different clusters are given different weights. This approach supports the comparison of a new project only with projects located in related clusters, based on the similarity measures. This method, like other methods where a subset is selected, deals with setting the correct value of k of the nearest project. Hihn et al. [19], described that the nearest neighbour method has significantly more outliers than Spectral Clustering does.

Lokan and Mendes [20], investigations showed that moving windows are helpful as a subset selection technique. Using 75 most recent projects for a new estimation makes this estimation more accurate than using all available data points. Amasaki and Lokan [21], later compare moving windows for ASEE and MLR models. They found MLR benefits more from windowing than analogy-based methods.

2. Problem statement

Historical projects in datasets, even if they are from identical organisation, should be dissimilar, which should negatively influence the model capability.

This research study addresses a problem of selecting a subset from historical datasets, which allows to construct a prediction model more relevant to newly estimated software development project. The inconsistence of historical dataset has negative impact to model performance.

Estimation performance and accuracy for the chosen subset selection techniques, together with an MLR model based on UCP variables, is compared.

Model initial configuration was derived from Silhavy et al. [11]. Silhavy et al., tested several MLR models by using a stepwise linear regression approach and the Best Performing Regression Model (BPRM) was selected. The following methods were evaluated and understood as subset selection techniques:

- k-means Clustering (k-means)
- Gaussian Mixture Model Clustering (GMM)
- Spectral Clustering (SC)
- Moving Window (MW)

These techniques were selected to demonstrate the differences between classical methods (k-means, GMM), and the modern one (SC), with alternative approach (MW). Selected clustering methods are favour for numerical data [22] and were previously used in the field [16,19]. MW method was previously studied by [23–25] an was introduced as subset selection method for increasing similarity among historical data.

In Silhavy et al. [11], the authors confirm that all UCP variables/features are significant for software development effort estimation.

Therefore, this study deals with UCP variables. The UCP method identifies:

- Unadjusted Actors´ Weights (UAW)
- Unadjusted Use Case Weights (UUCW)
- Technical Complexity Factors (TCF)
- Environmental Complexity Factors (ECF)

Other attributes available in datasets - including nominal attributes were not used, because nominal attributes, in general, cannot be used for new project classification to selected cluster.

The UCP method is based on assigning weights to clustered actors and use cases. It employs three cluster types: simple, average, and complex. The sum of the weighted actors creates a value called UAW; the UUCW value is calculated similarly. Two coefficients, technical factors and environmental factors, are used to describe the project, related information, and the experience level of the development team.

Actors play roles in the UAW variables [2,26]. A simple actor typically represents an application programming interface and a complex actor represents a human using a graphical user interface.

UAW and UUCW are derivate from UCM and the software size is based on actors or scenarios from use cases. TCF and ECF are used to describe the project, related information and the experience level of the development team. An Adjusted UCP (AUCP) score is obtained by summing the UAW and the UUCW, and then multiplying the resulting value by the TCF and ECF (1). See [11,27] for a detailed description of the UCP method.

$$AUCP = (UAW + UUCW) \times TCF \times ECF \tag{1}$$

2.1. Research questions and hypothesis formulation

RQ1: Can be a prediction ability of MLR models improved by using subset selection technique?

RQ2: Which subset selection technique is the best, when compared to UCP?

RQ3: Are moving windows equal to clustering techniques when prediction accuracy is compared?

Let us assume that, the Squared Prediction Errors (SE) of Tested Models (TM) - when the data subset technique is used, will be significantly lower than UCP's SE. To decide whether the model employing subset selection technique is more capable for prediction, a statistical hypothesis was tested:

H$_0$: $\mu SE_{TM} = \mu SE_{UCP}$; there is no difference in prediction capability between UCP and TM with subset techniques. No difference in the mean of such prediction errors.
Alternative hypothesis:
H$_1$: $\mu SE_{TM} < \mu SE_{UCP}$; there is difference in prediction capability between UCP and the TM with subset techniques. The mean of squared prediction errors is significantly lower for TM than for the UCP method.

This paper compares the accuracy of the tested models with that of the UCP method using a pair two sample *t*-test. The pair *t*-test for two samples is used as a test of the null hypothesis - that the means of two normally distributed populations (two sample) are equal. The *t*-test will be used for the evaluation of squared prediction errors. Usage of *t*-test was proofed by analysis in [11].

2.2. Evaluation criteria

All the tested methods were evaluated according to (2) Mean Absolute Percentage Error (MAPE), (3) the Sum of Squared Errors (SSE), (4) Mean Squared Error (MSE), (5) Root Mean Squared Error (RMSE) and (6) Normalised Mean Squared Error (NRMSE). MAPE was selected, because of in [28] authors proofs that MAPE has practical and theoretical relevance for evaluation of regression models and its intuitive interpretation in terms of relative error. Whereas Mean Relative Error (MRE) is not always optimal for describing the estimation accuracy [7]. MRE sometime omits overestimation, which lowers usability for software development effort estimation [29,30]. SSE is used because of its ability to represent errors for selected datasets. Cross-dataset comparability is weak (number of n must be handled), but overall it illustrates the model prediction performance.

MSE is seen as the best performing cost function in [8], which guide us to include it as possible performance measurement. RMSE and NRMSE are included for illustration purposes for future comparability of other models tested on given datasets. The equations are given as follows:

$$\mathrm{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{y_i} \times 100 \tag{2}$$

$$SSE = \sum_{i=1}^{n}\varepsilon_i^2 \tag{3}$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\varepsilon_i^2 \tag{4}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}\varepsilon_i^2}{n}} \tag{5}$$

$$NRMSE = \frac{RMSE}{n} \tag{6}$$

Where n is the number of observations, y_i is the known real value, $\hat{y}_i$ is the predicted value and ε is the prediction error value.

3. Methods used

3.1. K-means Clustering

K-means is a method for finding clusters in unlabelled data points. The number of clusters has to be set before the algorithm starts. This approach is based on finding data centres, where the initial centres are set randomly. The centres are iteratively moved to minimize data variance in one cluster [31]. The mean value of each feature is calculated, and those means are set as new cluster centres. In fact, it finds a subset for each centre for which the Cost Function (CF) of dissimilarity is minimized. In (7), the cost function in its generic form can be seen:

$$CF = \sum_{k=1}^{K}\sum_{x\in G_k} d(x, c_k) \tag{7}$$

Where, K represents the predefined number of clusters; G_k represents a cluster; d (8) is the distance/dissimilarity function; x is a vector; and c_k represents a cluster centre:

$$d(x, c_k) = (x - c_k)^T \times M(x - c_k) \tag{8}$$

Where, M is a distance matrix. The K-means approach does not guarantee that one will obtain an optimal solution. In many cases, a sub-optimal solution is found. There is a tendency in many iterative algorithms - including k-means, not to be convergent with the global minima. Therefore, it is used to run the k-means process several times – and, as result, a solution resulting from several runs is taken.

3.2. Gaussians Mixture Model Clustering

The Gaussian Mixture Model [32], (GMM), presents a model-based approach to clustering. GMM is understood as Probabilistic Clustering. In model-based clusters, a model for cluster and optimisation between data and model is researched. Each cluster is represented by its parametric distribution (component distribution). GMM can be used in advance in dominant pattern recognitions since these dominant patterns are related to the component distribution. (9), shows a multivariate Gaussian equation:

$$\mathcal{N}(x|\mu, \Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}}\exp\left\{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\Sigma)\right\} \tag{9}$$

Where, μ is, a component mean; Σ is the covariance matrix; and x represents a component. Mixing confidents can be interpreted by using (10). For GMM Clustering, the process is inverted and parameters like mixing confidents, means and covariance are researched:

$$p(x) = \sum_{k=1}^{K} p(k)p(x|k) \tag{10}$$

Then, a maximum likelihood function is used. The maximum of log-likelihood (11), is resolved by using an Expectation-Maximization (EM) algorithm, which describes an iterative scheme for the maximisation of the log-likelihood function. Firstly, the initial quest of the parameters is needed:

$$\ln\ \mathrm{p}(D|\pi, \mu, \Sigma) = \sum_{n=1}^{N} ln\left\{\sum_{k=1}^{K}\pi_k\mathcal{N}(\pi_k|\mu_k, \Sigma_k)\right\} \tag{11}$$

EM needs an e-steps approach for the parameters - which are latent variables and m-steps for the expected complete log-likelihood. In each iteration, the parameters are updated. The algorithm iterates until the log-likelihood of the data remains fixed. EM tends to stay within the local optimal value. Therefore, a new initial setting is recommended for running the algorithm again; and even several runs remain within the same optimum, and the global optimum might be found.

3.3. Spectral Clustering

The Spectral Clustering [33] algorithm is an Unsupervised Clustering method, based on a graphical representation, where each data point is a node and the edges between data points represent similarity, see Graph G (12). This represents a Degree Diagonal Matrix, in which a cell represents a sum of weights corresponding to each node from the graph - or respectively, a cell of matrix W:

$$G = (V, E) \tag{12}$$

Where, set V contains vertexes v_i and set E the edges e_i, which represent data points. Two vertexes are connected if the similarity s_{ij} between the corresponding data points $x_i$and x_j are larger or equal to the threshold; and the edge is weighted by s_{ij}. This means that part of the graph where edges with very low weights are found.

The K-nearest neighbour graph, ε-neighborhood graph and the fully-connected graph are typically used in Spectral Clustering [34]. The k-nearest neighbour graph connects v_i and $v_j$vertexes, where v_j is one of the k-nearest vertexes of v_i. The ε-neighborhood graph connects all data points where pairwise distances are smaller than ε. Later, the Adjacency Matrix W (13) is created:

$$W = (w_{ij}) \tag{13}$$

Where, $i, j = 1. .n$ and each cell in the matrix correspond to the edge' weight between two data points. If the weight is 0, then there is no connection between the edges. Finally, a Laplacian Matrix (14) is calculated:

$$L = D - W \tag{14}$$

Where, D is the diagonal matrix of the degree of vertex v_i. The L matrix is used for spectrum calculation - which is a key point in spectral clustering algorithms. The L matrix is used in un-normalized algorithms; when a normalized Laplacian algorithm is used, there are two possibilities (15) – a symmetric matrix and a random-walk:

$$L_{sym}: = D^{-1/2}LD^{-1/2} \ = \ I \ - \ D^{-1/2}WD^{-1/2}$$
$$L_{rw}: =D^{-1}L = I - D^{-1}W \tag{15}$$

The spectrum is a sorted list of the eigenvectors of a $L, L_{sym}\, or\, L_{rw}$ matrix. In fact, the eigenvectors represent a data-point of a data-set and an eigenvalue of a $L, L_{sym}\, or\, L_{rw}$ matrix. Spectral Clustering uses these eigenvectors as a feature. The clustering of features can be performed by any known algorithm. In this paper, the *k*-means algorithm is used.

3.4. Classifying into clusters

Multiclass Linear Discriminant Analysis, (MLDA), is used for the classification of new data-points to existing clusters. Each new project, which is represented as a new row in the testing set, is classified according to the clusters that were obtained from a training set. The testing set and training set must be matrices, with the same number of columns. In this case, more than two groups are used, which leads to seeking out more projections. Therefore, the *WP* (16) projection matrix is arranged as follows:

$$WP = W^T X \tag{16}$$

Where, W^T represents a Transposed Adjacency Matrix and X is the testing set data point matrix. The first step, (17), is used for the inter-class scatter matrix $\widehat{\Sigma_b}$ [35]:

$$\widehat{\Sigma_b} = \sum_{i=1}^{n} m_i(\bar{x}_i - \bar{x}) \ \times \ (\bar{x}_i - \bar{x})' \tag{17}$$

Where, m_i is the number of data points in the training set for each cluster; $\bar{x}_i$ is the mean for each cluster; and $\bar{x}$ is the mean vector. The linear transformation is then performed - where the key point is to maximize (17), and solve as a generalized eigenvalue problem.

The classification itself, is then performed in the transformed space by using metrics like Euclidean Distance or Cosine Distance. Newly-obtained instances are classified by minimizing a distance function with respect to a *k*th cluster centroid; and a mean value: $\bar{x}_k$.

3.5. Moving Window

Moving Window is a subset technique that can be understood in two types. The first type of MW is based on duration - and the second is based on the number of projects [20]. The Duration-based approach means, a training set project, a finished in a specific time frame, are used as parameters. If there is a large number of finished projects, then this is the most useful approach. Otherwise, if there is a low number of finished projects, then the “number of projects” is only an option.

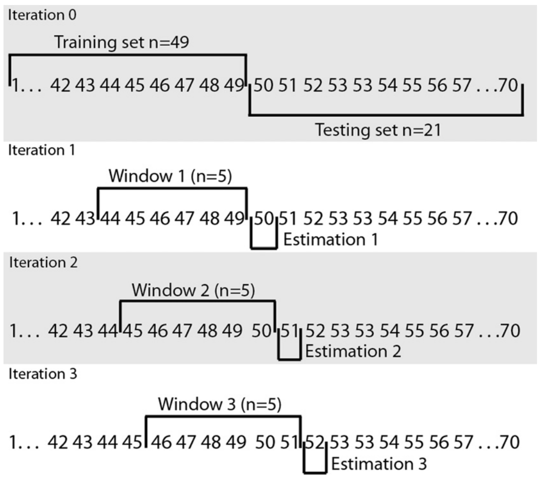

Fig. 1. Moving Window – project selection.

Fig. 1 illustrates the window function application is. Iteration 0 is described as an initial partitioning of a dataset - (n = 70), to a training set - (n = 49), and the testing part, (n = 21). The training dataset expands after a project is finished, (= data-point in testing set), and 5 most recent data-points are selected (see window).

In Iteration 1, the 5 most-recent projects, (44-49), are selected - and used to create the model. The model is used for the estimation of Project 50 - (the first one in the testing set).

In Iteration 2, Project 51 is estimated, and the model is constructed based upon Projects 45-49 from the training set and 50 from the testing set. This result, once again, in 5 the most recent projects. In Iteration 3, Project 52 is estimated, by means of a model which is based on Projects 46-49 from the training set and on Projects 50 and 51 from the testing set. In this illustrative example, it is expected that the projects are completed one by one.

If more than 1 project is finished before the next estimation is performed, then all finished projects are appended to the end of the training dataset.

Similarly, if more than 1 project must be estimated in each iteration a window can be used, which means that all projects are estimated by using an identical model.

Window-size represents the number of most recent projects in the training set. If window-size is n = 5, then the 5 most-recent data-points are used for the MLR model construction:

Authors of [36] recommend that the size of the training set, (the number of finished projects) should be around 30. Amasaki et al. [21] discuss windows sizes and conclude that 75 data points provides best available results. Window size depends on dataset size, this is a reason why in this study only smaller windows are tested. It is expected that the relevancy of earlier projects decreases over time. This claim can only be made if there is an assumption of increasing estimation accuracy, or if changes in size or project types are expected.

3.6. Stepwise Linear Regression

SLR is employed to select a BPRM by means of an automatic procedure. SLR is sometimes biased to the significance level in statistical assumptions. The best model is selected when the evaluation criteria are fulfilled. On the other hand, SLR should also help to reduce the number of predictors used in the selected model.

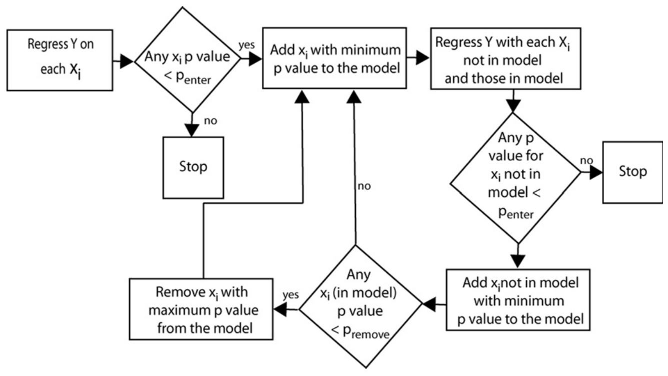
Regress Y on each X_i
Any x_i p value < p_enter
yes
no
Stop
Add x_i with minimum p value to the model
Regress Y with each X_i not in model and those in model
Any p value for x_i not in model < p_enter
no
Stop
Add x_i not in model with minimum p value to the model
Any x_i (in model) p value < p_remove
yes
no
Remove x_i with maximum p value from the model

Fig. 2. The Stepwise Regression Process.

Stepwise regression is based on the forward - and backward selection that involves an automatic process for the selection of independent variables; and can be briefly described as follows:

(1) Set a starting model, which contains predefined terms (backward); or set a null model (forward);
(2) Set limits for the final model—what type of model is needed?, whether linear terms are used?, squared terms?, or vice-versa?;
(3) Set an evaluation threshold, (p value of x, Sum of Squared Errors, or similar statistics) error is significantly decreased);
(4) Adding or removing terms; retesting the model;
(5) Stepwise regression halts when no further improvement in estimation occurs.

Fig. 2 depicts a schema of the stepwise process. There is a modification of forward selection - such that after each step in which a variable x is added, all the candidate variables in the model are checked to see whether their significance p has been reduced below a specified threshold p_{enter} (upper bound), or p_{remove} (lower bound). Forward selection starts as a null model and then iterates to add each variable which meet a condition. When a non-significant variable is found, it is removed from the model. Backward selection works in a similar manner, but removes variables when they are found to be non-significant. Therefore, stepwise regression requires two significance levels: the first, for adding variables; and the second, for removing variables.

SLR only is a method of building many models from a combination of predictors, therefore the multiple linear regression assumption has to be fulfilled. The multiple linear regression model, (18), is defined as:

$$y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \ldots + \beta_p X_{ip} + \varepsilon_i \tag{18}$$

Where, $i = 1, \ldots n$, y_i is the dependent variable; $X_{i1} \ldots X_{ip}$ are independent variables, (predictors); β_0 is an intercept; and, $\beta_1 \ldots \beta_n$ are regression coefficients. The value of ε_i represents the residuals. The model is designed as a matrix - where each row represents a data-point.

When MLR is a polynomial regression, (19); then the relationship between the dependent variables and the independent variable is modelled as an mth degree polynomial:

$$y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2}^2 + \ldots + \beta_p X_{ip}^m + \varepsilon_i \tag{19}$$

If ordinary Least Square Estimation is used, the vector of the estimated regression coefficients is in a matrix form - and can be defined as follows (20):

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T \boldsymbol{X})^{-1} \mathbf{X}^T \mathbf{y} \tag{20}$$

4. Experiment design

4.1. Project datasets

Methods, previously described above, were evaluated using two datasets. Dataset 1 - (DS1), was obtained from [2] – in which the dataset was based on [6] and [37]. Dataset 2 - (DS2), was collected by the authors and was first published - (in part), in [11].

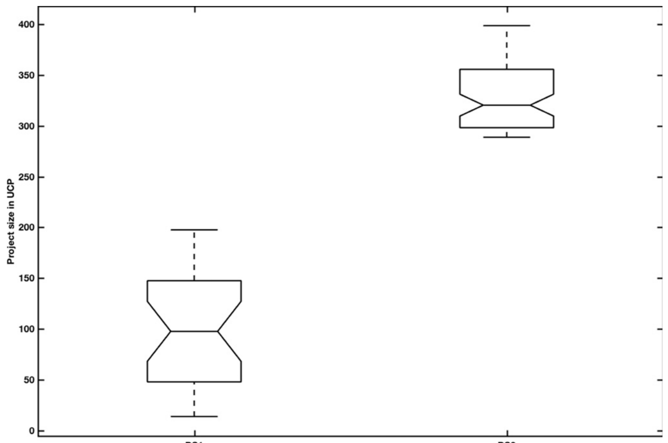

Fig. 3. Boxplots of project size, transformed into UCP.

For the purpose of this study, we do not aggregate data by data-donators or problem-domains. All projects were ordered by development completion date, for each donator. Fig. 3 shows a boxplot of the datasets. As shown, the two datasets represent groupings of different project sizes.

Table 1 shows the characteristics of the datasets used in the experiments. As shown, both datasets have a similar standard deviation, but DS2 contains larger projects. All the values in Table 1 are based on the Real_P20 [11], which describes the real project size in points (UCP). Datasets contain following variables:

- Project_No - project ID for identification purposes
- Simple Actors - Number of actor classify according UCP - simple actors
- Average Actors - Number of actor classify according UCP - average actors
- Complex Actors - Number of actor classify according UCP - complex actors
- UAW - Unadjusted Actor weight, computed by using UCP equation
- Simple UC - Number of use cases classified as simple - UCP number of steps is used Average UC - Number of use cases classified as average - UCP number of steps is used
- Complex UC - Number of use cases classified as complex - UCP number of steps is used
- UUCW - Unadjusted Use Case Weight - computed by using UCP equation
- TCF - Technical Complexity Factor
- T1-T13 - Technical factors significance
- ECF - Environmental Complexity Factors
- ENV1 - ENV8 - Environmental factors significance value
- Real_P20 - Real Effort in Person hours, divided by productivity factor (person hours per 1 UCP) (PF = 20)
- Real_Effort_Person_Hours - Real Effort (development time) in person-hours
- Sector - Problem domain of project
- Language - Programming language used for project
- Methodology - Development methodology used for project development
- ApplicationType - Classification of project type - provided by donator
- DataDonator - Anonymized acronym for data donator

In Table 1 Datasets' statistical characteristics can be seen. Median Person-Hours illustrates man value of project development time, which

Table 1
Datasets' characteristics.

	Median person-hours	Median Real_P20	Range Real_P20	SD Real_P20	Minimum Real_P20	Maximum Real_P20	n
Dataset 1	1952.500	97.625	183.650	57.063	13.850	197.500	28
Dataset 2	6406.000	320.300	109.750	33.212	288.750	398.500	70

was consumed from project's starting date to acceptance date. Median Real_P20 presents same value divided by PF = 20. It supposes that 20 person-hours corresponds to 1 UCP. This transformation was done because of data donators do not perform estimation using UCP.

Range Real_P20 describes difference between the smallest (Minimum Real_P20) and largest project (Maximum Real_P20). As can be seen variance of Dataset 1 is higher than for Dataset 2. Last column (n) stands for number of projects in each dataset.

4.2. Experiment procedure

For the purposes of this study, the BPRM [11] model is used. BPRM is built by SLR - with constraints applied. The model contains an intercept, linear terms, and squared terms. Both datasets - (DS1 and DS2), were divided in a ratio of 2:1, by using a hold-out method, which then was used to create training and testing sets. The testing sets are understood as time-ordered projects, which enter an estimation process. The training set was standardised (23) for clustering and used as non-standardised to obtain models for each subset/cluster, or as the source for the window range.

The experimental procedure for clustering algorithms is as follows:

(1) Creating training and testing sets by using the hold-out method
(2) Standardization of the training set, using the z-score method
(3) Selecting features for clustering (UAW, UUCW, TCF, ECF and Real_P20)
(4) Applying a clustering approach on the standardized training set
(5) Only solutions where each cluster contains more than 5 projects are selected
(6) Computing a BPRM for each of the clusters
(7) Projects in the testing set are classified into clusters
(8) Estimation of AUCP is performed by using a cluster-specific model
(9) PE for projects in the training set are computed
(10) SSE and other evaluation criteria are computed

The experiment procedure for the windows-based algorithm is as follows:

(1) Creating training and testing datasets by using the hold-out method
(2) The training set is understood as a historical project, in chronological order
(3) Computing a BPRM for projects in window-sizes: 5, 10 and 15
(4) Estimating a AUCP value for the first project in the training set
(5) Appending the project from Step 4 to the training set
(6) Repeating the procedure upon the testing set until it is empty
(7) PE for estimated projects in the training set are computed
(8) SSE and other evaluation criteria are computed

SLR - or generally, a MLR model; is sensitive to a minimum number of projects acceptable for training a model. When MLR is used in connection with clustering, a minimum number of projects must be set when a solution is selected. In this paper, only solutions where each cluster contains 5 or more data-points are used. This condition is used to obtain an appropriate level of machine precision when MLR models are calculated.

All algorithms tested in this experiment have no natural way as to how to apply this constraint, therefore the training set was clustered iteratively, with a predefined maximum value k, (21):

$$k = 2.\ .round\left(\frac{n}{5}\right) \tag{21}$$

Where, k is the number of clusters, and n is the number of projects in the training set. Clustering algorithms - (k-means, GMM clustering and Spectral Clustering), were used in combination with cosine similarity measurement (22). Cosine similarity was selected according to the results of the simulation study - [16], where several distance functions were tested.

$$d = 1 - \cos(x_i, x_j) \tag{22}$$

Where, x_i, x_j are feature sets that describe data-points in the training set. The k-nearest neighbour graph, was used for similarity measurement in Spectral Clustering; and k-means with cosine similarity were used for Eigenvectors Clustering – where a constraint of 5 data-points in each cluster was applied.

UAW, UUCW, TCF, ECF and Real_P20 were used as feature sets for all of the clustering algorithms. This feature set was selected because all of those attributes contribute to effort estimation. Using this set allows to cluster datasets not only according a project size, but also relationships among attributes can be included when the similarity is measured.

Training sets data were standardized. Standardizations were formed by using z-score. Z-scores measure the distance of a data point from the mean in terms of the standard deviation. This is also called standardization of data. The standardized data set has mean 0 and standard deviation 1, and retains the shape properties of the original data set (same skewness and kurtosis). Applying standardization equal weights are given to all of attributes, which are used in distance measurements.

$$z = \frac{(x - \mu)}{\sigma} \tag{23}$$

Where, x is a variable, μ stands for mean and σ represents a standard deviation.

5. Results

5.1. K-means Clustering

Only a solution in which a maximum of 3 clusters are created, can be accepted with respect to all constraints. The algorithm behaves the same for DS1 and DS2. Using more clusters is not possible, because of the initial constraint of 5 data-points in each cluster.

Table 2 shows that the 3-cluster solution allows one to achieve better prediction than a 2-cluster solution. Note: SSE for the DS2, 2 clusters: 2700.96 points; and 3 clusters: 289.36 only. Prediction Error is more than 9 times lower; when MAPE is compared, then 3 clusters are 3 times better than 2 clusters.

5.2. Spectral Clustering

In Table 3, the Spectral Clustering results are shown. Spectral Clustering achieves 5 clusters for DS2; and 3 clusters for DS1. The best DS2 results were achieved by 3 clusters. The SSE for 4 clusters is 552.04; and 5 clusters achieved: 3739.35. A similar difference is valid for all other criteria.

Table 2
K-means clustering results.

Number of clusters	SSE		MSE		RMSE		NRMSE		MAPE	
	DS1	DS2	DS1	DS2	DS1	DS2	DS1	DS2	DS1	DS2
2	96,237.00	2700.96	12,030.00	128.62	109.68	11.34	0.62	0.13	95.78	2.52
3	42,001.00	289.36	5250.10	13.77	72.46	3.71	0.41	0.04	53.05	0.97

Table 3
Spectral Clustering results.

Number of clusters	SSE		MSE		RMSE		NRMSE		MAPE	
	DS1	DS2	DS1	DS2	DS1	DS2	DS1	DS2	DS1	DS2
2	125,560.00	6451.49	15,695.00	307.21	125.28	17.53	0.71	0.20	69.50	2.30
3	42,001.00	289.36	5250.10	13.78	72.46	3.71	0.41	0.04	53.05	0.98
4	NaN	552.04	NaN	26.29	NaN	5.13	NaN	0.06	NaN	1.25
5	NaN	3739.35	NaN	178.06	NaN	13.34	NaN	0.15	NaN	2.39

Table 4
GMM Clustering results.

Number of clusters	SSE		MSE		RMSE		NRMSE		MAPE	
	DS1	DS2	DS1	DS2	DS1	DS2	DS1	DS2	DS1	DS2
2	51 773.00	2700.96	6471.70	128.62	80.45	11.34	0.45	0.13	96.58	2.52

5.3. *GMM Clustering*

In Table 4, GMM shows the worst clustering ability since only two clusters were identified for DS2, and no clusters for DS1. For DS1, this means that there was only one cluster with a majority of data-points and the second contains less than 5 data-points.

5.4. *Moving Window*

Moving Window, (Table 5), produced results which show its prediction ability - but also, error limits in its practical ability. Moving Window achieved a MAPE 27.26 for DS2, which means that the average prediction is about 27% lower - or higher, than real known value.

6. Discussion

In this study, four subset selection techniques methods were tested on the two datasets. Winning approaches (the best performing methods) are presented in Table 6, (DS1); and Table 7, (DS2). The best prediction ability was achieved by using k-means or spectral clustering, when 3 clusters for both datasets (DS1, DS2) were used. As can be seen from Table 3, Spectral Clustering is able to cluster a DS2 dataset into a maximum of 5 clusters - if the 5 data-point constraint is applied. The overall results show - (see Table 3) that 3 clusters outperform the 5-cluster solution. The winning application is Spectral Clustering, because it shown a better ability to cluster and if dataset is larger, than more clustering can be used if any minimal number of data-points in each cluster is requested.

For DS1, applying Spectral Clustering leads to decreases in errors by 40% as compared to UCP, and MAPE is 53%; other methods, (e.g. GMM, MW), are outperformed by SC or k-means. MW is outperformed by UCP.

When DS2 is evaluated; then similar results and behaviour can be observed. The results show yet again that SC is the winning solution – but in absolute values; all methods behave better than in DS1, which could be caused by higher of cluster consistency levels. This is driven by the source data - when DS2 contains more similar projects.

An interesting aspect can be seen in comparing DS2 and DS1 results. The difference between how all methods behave is clearly seen by using the MAPE criterion or by NRMSE - which are the only two criteria not biased regarding the number of data-points. As can be seen, the effects of clustering are higher for a bigger dataset (DS2).

Answering a RQ1 it can be shown, that UCP and BPRM (when identical MLR model is applied) are outperformed. Subselection technique for historical datasets impacted estimation accuracy. Applying clustering with 3 (k-means, SC) and 2 (GMM) clusters allows to increase a performance when compared to UCP or BPRM. MW, when window size equals 15 outperforms UCP but results are not significant for BPRM (Table 8b). The results were analysed using a “pair *t*-test”. This analysis revealed a significant difference between k-means and SC – as against UCP and BPRM. The test results at: $\alpha = 0.05$, can be seen in Table 8a and in Table 8b. The results are statistically significant for DS2 only. Fig. 4 illustrates the prediction errors for each data-point set in DS1 and DS2.

Table 5
Moving Window results.

Window size	SSE		MSE		RMSE		NRMSE		MAPE	
	DS1	DS2	DS1	DS2	DS1	DS2	DS1	DS2	DS1	DS2
5	135,760.00	1,537,772.33	1697.00	73,227.25	130.26	270.61	0.76	3.09	394.31	27.26
10	617,570.00	10,851,000.00	77,196.00	516,700.00	277.84	718.81	1.63	8.21	509.61	59.61
15	95,361.00	40,287.00	11,920.00	1918.50	109.18	43.80	0.64	0.50	333.98	9.35

Table 6
Selecting a winning approach for DS1.

Method	SSE	MSE	RMSE	NRMSE	MAPE
K-means - 3clust	42,001.00	5250.10	72.46	0.41	53.05
SC – 3clust	42,001.00	5250.10	72.46	0.41	53.05
GMM – 2clust	51,773.00	6471.70	80.45	0.45	96.58
MW – size 15	95,361.00	11,920.00	109.18	0.64	333.61
UCP	69,344.24	8667.60	93.10	0.54	165.52
BPRM [11]	29,499.90	3687.74	60.72	0.34	56.32

Table 7
Selecting a winning approach for DS2.

Method	SSE	MSE	RMSE	NRMSE	MAPE
K-means - 3clust	289.36	13.78	3.71	0.04	0.98
SC – 3clust	289.36	13.78	3.71	0.04	0.98
GMM – 2clust	2700.96	128.62	13.34	0.13	2.52
MW	40,287.00	1918.50	43.80	0.50	9.35
UCP	203,140.00	9673.20	98.35	1.12	26.82
BPRM [11]	7865.10	374.52	19.35	0.18	4.44

Table 8a
Statistical Significance of Tested Methods vs. UCP.

Method	DS1	DS2
k-means - 3clust	0.227	< 0.001
SC – 3clust	0.227	< 0.001
GMM – 2 clust	0.310	< 0.001
MW – size 15	0.661	< 0.001

Table 8b
Statistical Significance of Tested Methods vs. BPRM.

Method	DS1	DS2
k-means - 3clust	0.709	< 0.001
SC – 3clust	0.709	< 0.001
GMM – 2 clust	0.981	0.052
MW – size 15	0.849	0.963

The RQ2 asking about if all subset selection techniques are equal. As can be seen k-means and SC are only significantly better (see Tables 8a and 8b). GMM or MW cannot be understood as better when numerical nor statistical significance is studied.

The residuals show a significance that the MW approach is capable of predicting effort - when projects are natively similar. Whereas, clustering methods are less sensitive for changes in project types or size.

When RQ3 is discussed – if MW equals to clustering techniques. When MW are used together with MLR, they are not as capable as spectral clustering or k-means.

If all projects in a window are similar - and newly-predicted, this is significantly different that the MLR model and can provide an accurate prediction. Clustering methods excel because of the classification of projects in advance and because of using MLR models - which were constructed on similar projects.

Small window-size respects a practical point-of-view because it is usually not possible to use larger windows in practice; but does relate to historical dataset size. The influence of MW size can be seen in Fig. 5. Large windows produce better prediction - as previously confirmed in other studies [23–25]. Window sizes larger than 15 data-points bring significant improvements. But, SC is a more capable method. When SSE is compared, then SC achieves (for DS2): 289.36 vs 40,287.00; and MAPE is 0.98% compared to 9.35%. In Fig. 5, a development of SSE for a selected window size is shown. For better overview, the SSE is shown on a log-transformed *y* axis; while MAPE is shown on a normal *y*-axis (percentage).

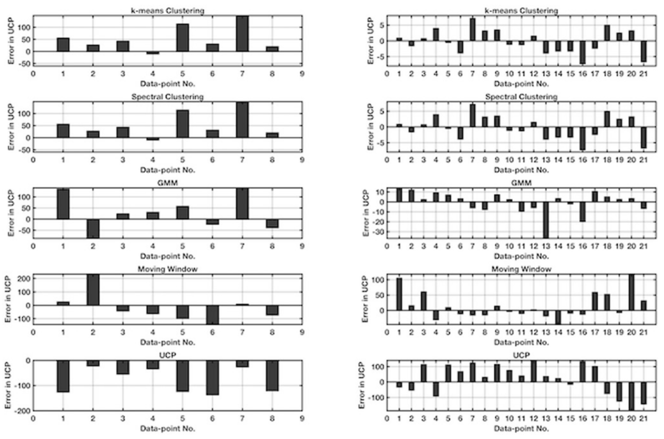

Fig. 4. Residuals for individual data-points (DS1 – left, DS2 – right).

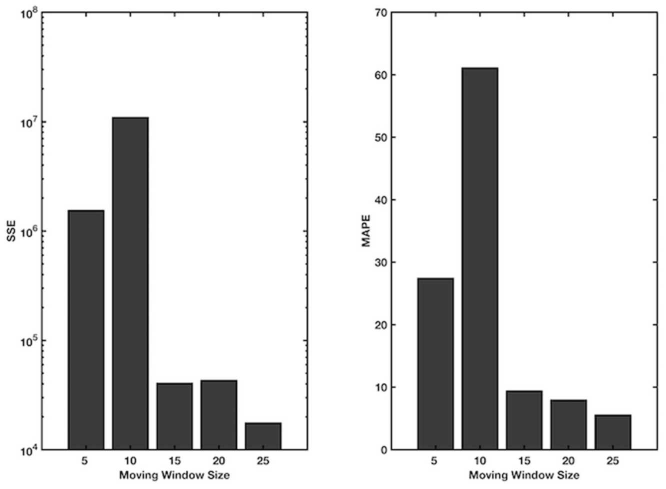

Fig. 5. Moving Window Size Impact on SSE and MAPE (on DS2).

6.1. Threats to validity

In this study, two datasets - which both are publicly available, are used. Both datasets are related to the UCP estimation approach, which makes it a bias-to-estimation method. All conclusions are valid for the experiment setup presented herein. No other datasets for UCP methods are available, therefore only mentioned datasets are used.

A dataset is used in the hold-out setup, which allows one to divide it into training and testing sets. Hold-out is only an option due to limited size of DS1. This division is based on random generation; therefore, replication is limited if the same random generator configuration is not used.

Sub-selection Techniques, (Clustering), are sensitive to initial seed setup. Seed is used for initial centroid selection for clustering methods.

The rule of a minimum of 5 observation (historical data-points) in each cluster is applied. This rule allows to avoid creation of too many small clusters, which an impropriate for MLR model construction.

In the MW approach, the windows size takes on an important role. In this study, a window of size of 15 is used because the DS1 is small and larger windows cannot be tested.

7. Conclusion

The effect of subset selection techniques was studied, and three clustering techniques: (k-means, SC, GMM), as well as the MW method, were evaluated against the UCP method. All techniques were used as data-preparation methods for the MLR models. Clustering allows selecting more similar projects - which leads to better estimation, because of more capable is used for prediction. MW is similar in performance, only if a larger size is used, (15 data-points). This means MW is not acceptable for smaller data-sets. MW is sensitive to significant changes in project-size, or characteristics. The MW solution is not a robust solution, due to its data sensitivity.

When RQ1 is discussed, it can be concluded that Subset Selection Techniques improve overall prediction and MLR performance is significantly improved. GMM Clustering provides small improvements - when compared to UCP or BPRM, (for DS1). MW - when window-size equals 15, is better than UCP; but, smaller window sizes are not as capable.

In RQ2, we can declare that the best performing subset selection technique is SC, which we can assume is the most influential method. In this study, SC performance is equal to k-means; but, as can be seen in Table 3, SC is able to cluster data better, (into more clusters), if the dataset is larger. This is supported by the results, when DS2 is employed.

To answer RQ3; it can be concluding that MW is usable – only for larger dataset, but is still not as good as SC when windows size is large - (15 data-points). Otherwise, MW caused rises in prediction errors. MW is sensitive to changes in project characteristics; if a newly predicted project is dissimilar to those in the historical dataset, then MW can introduce a significant error.

In conclusion clustering methods reduce the prediction error of linear regression methods significantly, where SC is a winning method of subset selection techniques. SC can reduce a prediction error by up to 98%, when compared to UCP and by 30% when compared to BMRM [11]. MW produces inconsistent results and it is a data sensitive method.

In future research, a hybrid method - which will combine a MW and SC approach, will be investigated and weighted windows based on windows function will be evaluated.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.infsof.2017.12.009.

References

[1] G. Karner, Metrics For Objectory', Diploma, University of Linkoping, Sweden, December 1993, p. 21 No. LiTH-IDA-Ex-9344.
[2] R Silhavy, P Silhavy, Z Prokopova, Algorithmic optimisation method for improving use case points estimation, PloS ONE 10 (11) (2015) e0141887.
[3] M. Ochodek, B. Alchimowicz, J. Jurkiewicz, J. Nawrocki, Improving the reliability of transaction identification in use cases, Inf. Softw. Technol. 53 (8) (2011) 885–897, http://dx.doi.org/10.1016/J.Infsof.2011.02.004 PubMed PMID: WOS:000292176300007.
[4] A.B. Nassif, D Ho, L.F. Capretz, Towards an early software estimation using log-linear regression and a multilayer perceptron model, J. Syst. Softw. 86 (1) (2013) 144–160.
[5] V. Anandhi, R.M. Chezian, Regression techniques in software effort estimation using cocomo dataset, International Conference on Intelligent Computing Applications (Icica 2014), 2014, pp. 353–357, , http://dx.doi.org/10.1109/Icica.2014.79 PubMed PMID: WOS:000358253500072.
[6] M. Ochodek, J. Nawrocki, K. Kwarciak, Simplifying effort estimation based on use case points, Inf. Softw. Technol. 53 (3) (2011) 200–213, http://dx.doi.org/10.1016/j.infsof.2010.10.005.
[7] M. Jorgensen, Regression models of software development effort estimation accuracy and bias, Empir. Softw. Eng. 9 (4) (2004) 297–314, http://dx.doi.org/10.1023/B:EMSE.0000039881.57613 cb. PubMed PMID: WOS:000224569200003.
[8] T. Urbanek, Z. Prokopova, R. Silhavy, V. Vesela, Prediction accuracy measurements as a fitness function for software effort estimation, SpringerPlus 4 (2015) 17, http://dx.doi.org/10.1186/s40064-015-1555-9 PubMed PMID: WOS:000368718000002.
[9] M. Jorgensen, M. Shepperd, A systematic review of software development cost estimation studies, IEEE T Softw. Eng. 33 (1) (2007) 33–53, http://dx.doi.org/10.1109/Tse.2007.256943 PubMed PMID: WOS:000242312200003.
[10] J.F. Wen, S.X. Li, Z.Y. Lin, Y. Hu, C.Q. Huang, Systematic literature review of machine learning based software development effort estimation models, Inf. Softw. Technol. 54 (1) (2012) 41–59, http://dx.doi.org/10.1016/j.infsof.2011.09.002 PubMed PMID: WOS:000297871500003.
[11] R. Silhavy, P. Silhavy, Z. Prokopova, Analysis and selection of a regression model for the use case points method using a stepwise approach, J. Syst. Softw. 125 (2017) 1–14 http://dx.doi.org/10.1016/j.jss.2016.11.029.
[12] A. Idri, F.A. Amazal, A. Abran, Analogy-based software development effort estimation: a systematic mapping and review, Inf. Softw. Technol. 58 (2015) 206–230, http://dx.doi.org/10.1016/j.infsof.2014.07.013 PubMed PMID: WOS:000347022800012.
[13] A. Nassif, M. Azzeh, L. Capretz, D. Ho, Neural network models for software development effort estimation: a comparative study, Neural Comput. Appl. (2015) 1–13, http://dx.doi.org/10.1007/s00521-015-2127-1.
[14] M. Azzeh, A.B. Nassif, A hybrid model for estimating software project effort from use case points, Appl. Soft Comput. (2016).
[15] V.K. Bardsiri, D.N.A. Jawawi, S.Z.M. Hashim, E. Khatibi, Increasing the accuracy of software development effort estimation using projects clustering, Iet Softw. 6 (6) (2012) 461–473, http://dx.doi.org/10.1049/iet-sen.2011.0210 PubMed PMID: WOS:000310517200001.
[16] Z. Prokopova, R. Silhavy, P. Silhavy, The effects of clustering to software size estimation for the use case points methods, Adv. Intell. Syst. Comput. (2017) 479–490.
[17] V.K. Bardsiri, D.N.A. Jawawi, S.Z.M. Hashim, E. Khatibi, A flexible method to estimate the software development effort based on the classification of projects and localization of comparisons, Empir. Softw. Eng. 19 (4) (2014) 857–884, http://dx.doi.org/10.1007/s10664-013-9241-4 PubMed PMID: WOS:000336388500003.
[18] J. Kennedy, R. Eberhart, Particle swarm optimization, 1995 IEEE International Conference on Neural Networks Proceedings, 1–6 1995, pp. 1942–1948, , http://dx.doi.org/10.1109/Icnn.1995.488968 PubMed PMID: WOS:A1995BF46H00374.
[19] J. Hihn, L. Juster, J. Johnson, T. Menzies, G. Michael, Improving and expanding NASA software cost estimation methods, Aerospace Conference, 2016 IEEE, IEEE, 2016.
[20] C. Lokan, E. Mendes, Applying moving windows to software effort estimation, Int. Symp. Emp. Softw. (2009) 111–122 PubMed PMID: WOS:000274866100011.
[21] S. Amasaki, C. Lokan, The effects of moving windows to software estimation: comparative study on linear regression and estimation by analogy, Proceedings of the 2012 Joint Conference of the 22nd International Workshop on Software Measurement and the 2012 Seventh International Conference on Software Process and Product Measurement (Iwsm-Mensura 2012), 2012, pp. 23–32, , http://dx.doi.org/10.1109/Iwsm-Mensura.2012.13 PubMed PMID: WOS:000317102600006.
[22] A. Saxena, M. Prasad, A. Gupta, N. Bharill, O.P. Patel, A. Tiwari, et al., A review of clustering techniques and developments, Neurocomputing 267 (Supplement C) (2017) 664–681 https://doi.org/10.1016/j.neucom.2017.06.053.
[23] C. Lokan, E. Mendes, Investigating the use of moving windows to improve software effort prediction: a replicated study, Empir. Softw. Eng. 22 (2) (2017) 716–767, http://dx.doi.org/10.1007/s10664-016-9446-4 PubMed PMID: WOS:000399891400004.
[24] S. Amasaki, C. Lokan, Evaluation of moving window policies with CART, Int. Worksh. Empir. Eng. (2016) 24–29, http://dx.doi.org/10.1109/Iwesep.2016.10 PubMed PMID: WOS:000381744800005.
[25] S. Amasaki, C. Lokan, On applicability of fixed-size moving windows for ANN-based effort estimation, Proceedings of 2016 Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement (Iwsm-Mensura), 2016, pp. 213–218, , http://dx.doi.org/10.1109/IWSM-Mensura.2016.31 PubMed PMID: WOS:000399139200029.
[26] M. Azzeh, A. Nassif, S. Banitaan, F. Almasalha, Pareto efficient multi-objective optimization for local tuning of analogy-based estimation, Neural Comput. Appl. (2015) 1–25, http://dx.doi.org/10.1007/s00521-015-2004-y.
[27] R. Silhavy, P. Silhavy, Z. Prokopova, Applied least square regression in use case estimation precision tuning, Software Engineering in Intelligent Systems, Springer International Publishing, 2015, pp. 11–17.
[28] A. de Myttenaere, B. Golden, B. Le Grand, F. Rossi, Mean absolute percentage error for regression models, Neurocomputing 192 (Supplement C) (2016) 38–48 https://doi.org/10.1016/j.neucom.2015.12.114.
[29] B.A. Kitchenham, L.M. Pickard, S.G. MacDonell, M.J. Shepperd, What accuracy statistics really measure [software estimation], Softw. IEE Proc. 148 (3) (2001) 81–85, http://dx.doi.org/10.1049/ip-sen:20010506.
[30] M. Shepperd, M. Cartwright, G. Kadoda, On building prediction systems for software engineers, Empir. Softw. Eng. 5 (3) (2000) 175–182, http://dx.doi.org/10.1023/a:1026582314146.
[31] G. James, D. Witten, T. Hastie, R. Tibshirani, An Introduction to Statistical Learning, Springer, 2013.
[32] D. Reynolds, Gaussian Mixture Models, in: SZ Li, A Jain (Eds.), Encyclopedia of Biometrics. Boston, MA: Springer US, 2009, pp. 659–663.
[33] U. von Luxburg, A tutorial on spectral clustering, Stat. Comput. 17 (4) (2007) 395–416, http://dx.doi.org/10.1007/s11222-007-9033-z.
[34] M. Soltanolkotabi, E. Elhamifar, E.J. Candes, Robust subspace clustering, Ann. Stat. 42 (2) (2014) 669–699, http://dx.doi.org/10.1214/13-Aos1199 PubMed PMID: WOS:000336888400014.
[35] T. Li, S. Zhu, M. Ogihara, Using discriminant analysis for multi-class classification: an experimental investigation, Knowl. Inf. Syst. 10 (4) (2006) 453–472, http://dx.doi.org/10.1007/s10115-006-0013-y.
[36] B. Kitchenham, S.L. Pfleeger, B. McColl, S. Eagan, An empirical study of maintenance and development estimation accuracy (vol 64, pg 57, 2002), J. Syst. Softw. 74 (2) (2005) 227, http://dx.doi.org/10.1016/j.jss.2004.07.010 PubMed PMID: WOS:000224874700010.
[37] A Subriadi, P Ningrum, Critical review of the effort rate value in use case point method for estimating software development effort, J. Theoretical Appl. Inf. Technol. 59 (3) (2014) 735–744.