Optainet-based technique for SVR feature selection and parameters optimization for software cost prediction

Assia Najm[1*], *Abdelali Zakrani*[2], and *Abdelaziz Marzak*[1]

[1]Department of Mathematics & Computer Sciences, FSB M'sik, Hassan II University, Casablanca, Morocco
[2]Department of Computer Science Engineering, ENSAM, Casablanca, Morocco

Abstract. The software cost prediction is a crucial element for a project's success because it helps the project managers to efficiently estimate the needed effort for any project. There exist in literature many machine learning methods like decision trees, artificial neural networks (ANN), and support vector regressors (SVR), etc. However, many studies confirm that accurate estimations greatly depend on hyperparameters optimization, and on the proper input feature selection that impacts highly the accuracy of software cost prediction models (SCPM). In this paper, we propose an enhanced model using SVR and the Optainet algorithm. The Optainet is used at the same time for 1- selecting the best set of features and 2-for tuning the parameters of the SVR model. The experimental evaluation was conducted using a 30% holdout over seven datasets. The performance of the suggested model is then compared to the tuned SVR model using Optainet without feature selection. The results were also compared to the Boruta and random forest features selection methods. The experiments show that for overall datasets, the Optainet-based method improves significantly the accuracy of the SVR model and it outperforms the random forest and Boruta feature selection methods.

1 Introduction

One of the important challenges for any software house (or company) is to achieve a good and healthy return on investment. This can be controlled by using an efficient allocation of resources and perfect management of the project's plans and budgets. To that end, the prediction of software development cost remains crucial. Several methods have been proposed in the literature such as support vector regression (SVR) [1], artificial neural networks [2], and decision trees (DT) [3], [4], etc. However, it is difficult or even impossible to find a technique that performs well in all situations. The performance of any SCPM depends greatly on the dataset characteristics (outliers, size, missing values, categorical attributes), and also on the suitable optimization technique for tuning the model hyperparameters. Therefore, performing simultaneous feature selection (FS) and parameter

---

[*] Corresponding author: assia.najm@gmail.com

optimization (PO) will enhance the accuracy of SDCPM. Principally, the FS preprocessing step will make the data purer by removing unimportant features [5], while the PO step will find the best configuration that enhances the performance of the used SDCPM. Generally, FS methods are grouped into three categories: the embedded, filter, and wrapper technique. In this paper, we used the wrapper method for FS because of its accuracy.

Consequently, in this study, we used the Optainet algorithm [6] simultaneously for FS and PO before building the SVR model. Especially, we use ε-SVR-RBF, so we are faced with two problems: the first is related to the way of selecting relevant attributes, and the second concerns the choice of appropriate parameters (the model complexity B, ε, and kernel parameters λ). As claimed by Yang and Honavar [7], the selection of pertinent features has many advantages on the effort, the model accuracy, the learning time, and the number of samples required for learning. Similarly, the suitable hyperparameters setting(B, λ, ε) may enhance the accuracy of the SVR model. Unlike Grid Search (GS) algorithm which performs only the parameter optimization, the Optainet algorithm can perform FS and parameter optimization at one time. The objective of this paper is to improve the accuracy of the SVR model using an Optainet algorithm. The resultant model is then referred to as SVR_Optainet_FS_PO which signifies SVR with FS and PO, and then we compare it to the four other SVR models which are as follows: Our second developed model: the SVR_Optainet_PO which means SVR model with Optainet for only PO without FS step, and three other models described in [8]: the SVR with backward feature elimination (SVR_BFE), SVR with Boruta FS method (SVR_Boruta), and SVR without FS (SVR).

The organization of this study is divided into six sections: Section 2 lists the related works of SVR models dealing with SDCP. Section 3 explains the overall design of the proposed model. Section 4. presents the used datasets, accuracy measures, and validation method, 5 presents all the obtained findings. Finally, section 6 summarizes this study and gives insights into future work.

2 Related works

We intend from this section to present a summary of some identified studies that investigates the use of SVR for SDCP:

Oliveira [1] investigated the use of the grid search technique in optimizing the parameters of two SVR models: the SVR with RBF and SVR with linear kernels. The results show that SVR performs better than Linear and RBFNs regressors. The experimentation was made using only one dataset, the NASA dataset, and we note that no FS method was employed.

Besides, the SVR model was also investigated in these papers [9, 10], where they used a genetic algorithm (GA) for both FS and PO. They proposed a binary chromosome that represents the suggested hyperparameters and the selected features. Empirical experimentation shows that the accuracy of SVM is enhanced using the GA-based method, and it outperforms the bagged M5P and the Bagged MLP.

In [11], the authors proposed a PSO-based approach For FS and PO of the SVR model. Nevertheless, this paper made the evaluation using only one dataset which is the Desharnais dataset. Also, the authors didn't employ the commonly used criteria in SDCPM.

In [12]–[14] the authors also investigate the use of the SVR model for estimating the cost of web projects over the TUKUTUKU dataset. The results confirm that SVR is a promising technique thanks to its better performance regarding other SDCPM. They claimed that is suitable for cross-company datasets through its kernels and parameter settings.

The authors in [15] also proposed an improved SVR model based on the tabu search (TS)method to tune the SVR parameters. The experimentation was performed using

TUKUTUKU and PROMISE repository datasets. The main finding is that SVR using TS achieves a great improvement compared to CBR and stepwise regression.

3 The proposed model

The present section gives the overall view of the SVR model, especially the SVR_Optainet_FS_PO that refers to the SVR model using RBF kernel and Optainet FS and parameter optimization.

3.1 SVR model

The support vector machine (SVM) is an encouraging machine-learning (ML) model that has offered valuable results in both classification and regression problems [16]. The SVM approach has several attractive features like the sparse way of presenting solutions, good ability of generalization, besides the capacity of avoiding local minimums thanks to the structural risk minimization concept.

In this paper, we employ ε-SVR that introduces the ε-insensitive loss function (see Fig. 1). This means that all errors less than a defined threshold (inside bars in the figure) are neglected, while errors induced by the points located out of the bars are calculated using ψ and ψ^* like in Fig1.

In the case of nonlinear regression, the next (Eq. 1), is used where φ denotes a nonlinear function that maps the low input space to the high output space; w represents the weights vector, and c the threshold.

$$h(x)=w^{T} \cdot \varphi(x) + c \qquad (1)$$

We mention that w and c are selected for optimizing the next problem [17]:

$$\text{Minimizing}_{w,b,\psi,\psi^*} \qquad \frac{1}{2} < w, w > \; + B \sum_{i=1}^{l} (\Psi_i + \Psi_i^*)$$

$$(< w, \varphi(x_i) > + c) - y_i \leq \varepsilon + \psi_i, \qquad (2)$$

$$\text{Subjects to} \qquad y_i - (< w, \varphi(x_i) > + c) \leq \varepsilon + \Psi_i^*,$$

$$\psi_i, \Psi_i^*, \geq 0.$$

The ε represents the function's deviation, the regularization parameter B reflects the trade-off between the error's tolerance above ε and the flatness of h, the ψ, ψ^* as proposed in[18] presents the slack variables that define the tolerated deviations above the error ε. The base mechanism of the SVR model is minimizing the objective function which takes into account the norm of w and the loss of ψ_i, *and* Ψ_i^*as expressed in (Eq. 2).

We note that The Lagrangian multiplier is used in the SVR method, it is related to the dot product of φ (x). It can be realized through the kernel function determined as:

$H(x_i, x_j) = <\varphi (x_i), \varphi (x_j)>$ which prevents the explicit computation of the φ (x). We refer the reader to [17] for more details.

In the present study, we used the SVR-RBF thanks to its ability to generate good estimates [19], [20]. The computation of the SVR-RBF kernel is done via this expression:

$H(x_i, x_j) = \exp(-\lambda \| x_i - x_j \|^2)$. Hence, parameter λ should be cautiously chosen beside the B and the ε parameters.

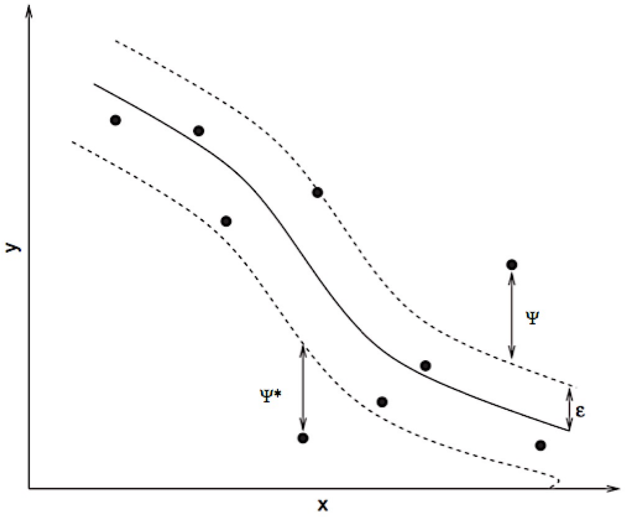

Fig. 1. The ε-SVR regression.

3.2 Optainet algorithm

The artificial immune network (aiNet) model was introduced by de Castro and Zuben [21], it is a graph composed of nodes (antibodies) and edges. It is essentially based on the immune system (IS) theory [22] where the IS generates a great number of antibodies or attempts to provide the best-suited antibodies for attacking antigens. The modelization of this phenomenon corresponds to a function optimization process.

Many algorithms inspired by the biological domain and especially by the IS were used to achieve global optimization like the opt-IA [23], the B-Cell algorithm[24], the opt-aiNet[6], and the Clonalg algorithm[25].

In this work, the SVR-RBF with Optainet optimization [6] is used to optimize the SVR-RBF parameters. The Optainet explores the effectiveness of the aiNet theory, which simulates the human immune system activity (it has a robust memory, and a great ability to differentiate the self cells from the foreign ones). The Optainet algorithm has the following advantages:

- Inclusion of stopping criteria.
- Ability to check and maintain various optimal solutions.
- Capacity to exploit and explore the whole space of search.
- The capability of dynamic adjustment for the population size.

We adopt the next terminology to explain the Optainet algorithm:

Cells	each cell in the network is composed of population values. In the Euclidean space, it is denoted with a multivalued vector.
Cell's Fitness	stands for the objective function value using a particular network cell.
Cells Affinities	represents the value of distance (Euclidean) that exists between two cells.
Cloning	cloning is producing replication of original cells (or parents).
Mutation	The generated copies (or offspring) will be mutated to be different from their parents (mutation).

The overall Optainet steps are described as noted below:

BEGIN

1) Randomly initialize the population.
2) If the stopping criteria are not satisfied, do:
    a. Calculating the fitness of each cell and evaluating it according to the used objective function. Then proceed to its normalization.
    b. Performing cell cloning according to the chosen Nc, which represents the number of offsprings (or clones).
    c. Performing the mutation operation, where each clone will be inversely proportional to its parent fitness value as expressed below:

$$cl' = cl + \sigma N(0,1) \qquad (3)$$
$$\sigma = (1/\eta) exp(-h^*)$$

where N(0,1) presents the random Gaussian variable with standard deviation equal to 1 and a zero mean, cl is the parent cell, and the cl' is the cell resulting from the mutation of cl, also $h^* \in [0,1]$ represents the normalized fitness of a cell, while the parameter η controls the exponential function's decay. Acceptable mutations are those that are within the domain interval.

    d. Calculating the fitness of each cell present in the population (including cloned and mutated ones).
    e. Selecting the highest fitness cells per clone, and excluding the others.
    f. Evaluating all cells affinities, then suppressing the cells having affinities lower than a predetermined suppression minimum σ_s (threshold).
    g. Back into the second step, after adding a p% of random cells.
3) Else, selecting the highest fitness cell and Ending the algorithm.

END

3.3 Optainet-based SVR model

This subsection describes the design of the cells' network, the fitness function, and the design of the suggested Optained-based SVR feature selection and parameter optimization.

3.3.1 Cell Design

We used the ε-SVR model with RBF kernel to implement the proposed model. So, The parameters (B, λ, ε), and the dataset's features should be cautiously chosen, and carefully optimized using our suggested Optainet-based model. For that reason, each cell in the network includes four elements: the B, λ, ε, and features bits. Figure 2 shows the cell design representation. The three first parts represents the SVR parameters while the last part represent the features bits relative to each dataset. We mention that the feature bits include 0, or 1 where 0 signifies that the attribute is unimportant (not selected) while 1 reflects the importance of the attribute (is selected).

B_i	λ_j	ε_k	f_1	$\dots . f_\alpha \dots ..$	f_N

Fig. 2. The cell with four parts, the B, λ, ε, and the features bits.

3.3.2 Fitness function

To evaluate the performance of SDCPM, various criteria were suggested and employed in the literature. Especially, in this study, we used three measures (Pred, MdMRE, and MMRE) as detailed in section 4.2 which represents the frequently used ones in many studies [26]. For example Braga et al. [27], Shin and Goel [28], etc. An acceptable value of Pred (25%) is more than or equal to 75% while for MMRE, and MdMRE values must be less than or equal to 0.25.

Therefore, in this study, we employ a fitness function based on these three criteria (see Eq. 4) where we look for cells with higher values of Pred and small values of MMRE, MdMRE

$$Fitness\_function=(100-Pred(25\%))+(MMRE+MdMRE). \quad (4)$$

3.3.3 Optained-based model Design

We describe in this subsection the architecture of our model for feature selection and parameter optimization. Figure 3 shows the main steps of our proposed models that are explained next:

Data scaling: We firstly proceed to the data-preprocessing phase, where we normalize all the available data in the range between 0 and 1 to strengthen the accuracy of the regressor. The expression used for normalization is noted below:

$$y^* = \frac{y - y_{min}}{y_{max} - y_{min}} \quad (5)$$

where y_{max} is the maximal data value in the dataset, y_{min} is the minimal data value, y is the original value, and y^* is the value after normalization.

After that, the main steps of optainet akgorithm were performed : the Cloning, mutation, fitness evaluation, affinity measurement, and network suppression (see section 3.2). The output of this algorithm is finding the best solution with feature selection and parameter optimization.

The Optainet algorithm seeks the optimal SVR configuration by exploring the parameters search space provided in Table 1. This is done by respecting the fitness function. Hence, the optimal solution that maximizes the fitness function is then chosen.

Table 1.The Values of SVR Model & Optainet Algorithm

Methods	**Hyperparameters**

Optainet	Number of iteration ={ 500 }
	Size of population ={ 50 }
	Number of clones ={ 20}
	The threshold of suppression ={0.8}
	Newcomers percentage ={ 50%}

SVR	Type= {eps-regression}
	Kernel function = { RBF }

	Range values of Hyparameters

	Complexity = [1000, 1500, 3000, 4000, 5000]
	Kernel parameter =[0.9,1,2]
	Epsilon = [0.01, 0.02,0.03]

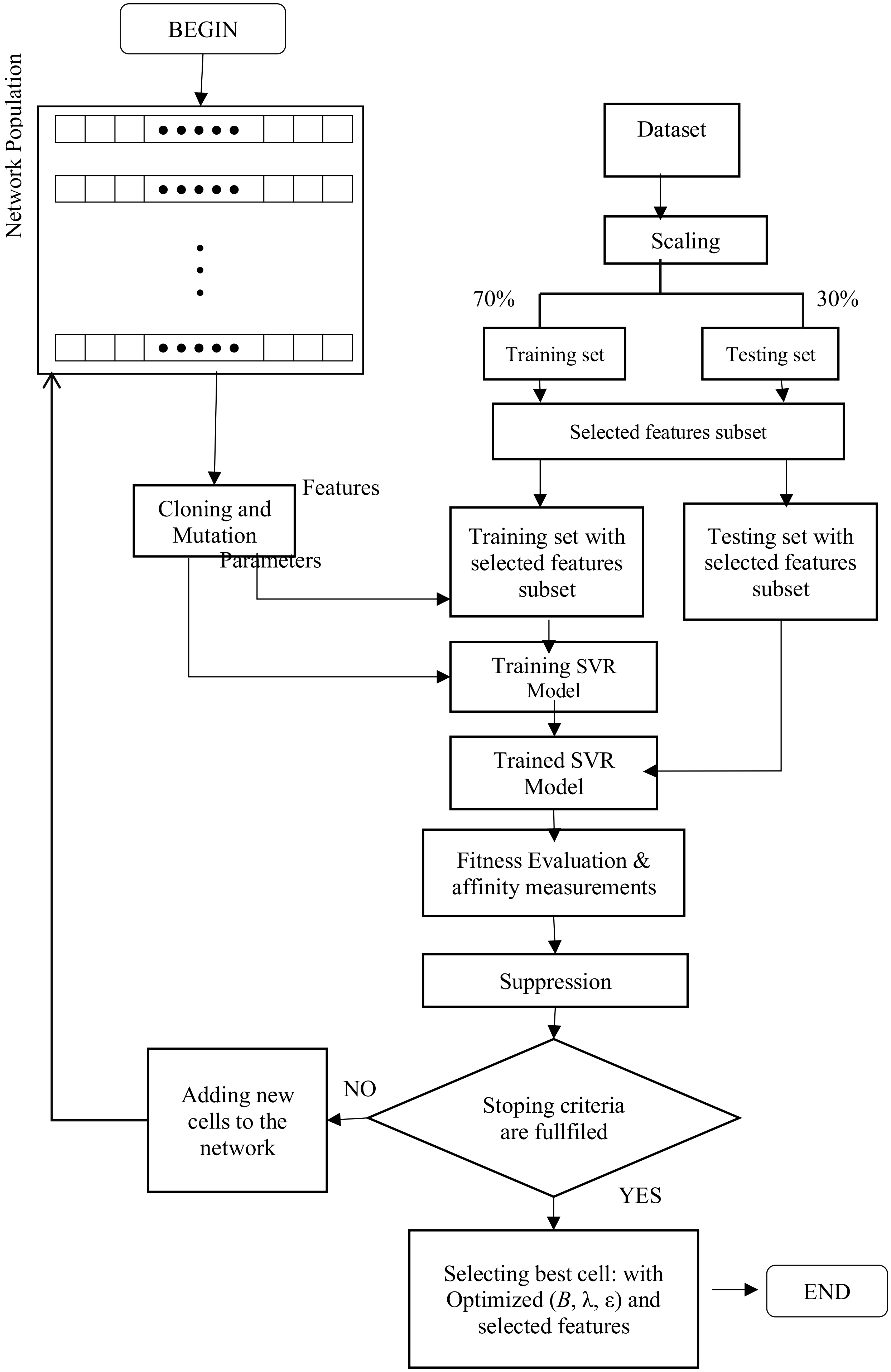
BEGIN
Network Population
Dataset
Scaling
70%
30%
Training set
Testing set
Selected features subset
Features
Cloning and Mutation
Parameters
Training set with selected features subset
Testing set with selected features subset
Training SVR Model
Trained SVR Model
Fitness Evaluation & affinity measurements
Suppression
Stoping criteria are fullfiled
NO
YES
Adding new cells to the network
Selecting best cell: with Optimized (B, λ, ε) and selected features
END

Fig. 3. Design of the SVR_Optainet_FS_PO model.

4 Experimental design

In this section, we describe the used datasets, the performance metrics, and the validation techniques.

4.1 Datasets

We used seven well-known datasets. Notabely, the Albrecht, Kemerer, Miyazaki, Desharnais, Cocomo, ISBSG, and China. A detailed description of these datasets is presented in Table 2. This table shows each dataset size, number of attributes, the maximum, the minimum, the median, the mean, the effort's skewness, and kurtosis.

Table 2.Datasets Descriptive Statistics

Dataset	Size	Nb. Features	Effort						
			Min	Max	Mean	Std	Median	Skewness	Kurtosis
Albrecht	24	7	0,50	105,20	21,88	28,42	11,45	2,30	4,67
COCOMO	252	13	6,00	11400,00	683,44	1810,62	98,00	4,39	20,51
China	499	15	26,00	54620,00	3921,05	6480,86	1829,00	3,93	19,31
Desharnais	77	8	546,00	23940,00	4833,91	4188,19	3542,00	2,04	5,30
ISBSG	151	6	24,00	60270,00	5039,13	8471,11	2449,00	4,17	21,10
Kemerer	15	6	23,20	1107,31	219,25	263,06	130,30	3,07	10,59
Miyazaki	48	7	5,60	1586,00	87,48	228,76	38,10	6,26	41,36

4.2 Accuracy measures

The accuracy measures are used to evaluate and compare the performance of the SCPM. For this purpose, we used three commonly used measures (see Eq. 7-9).

Mainly, the first is the mean magnitude of relative error (*MMRE*) (see Eq. 7) which represents the average of the magnitude of relative error (*MRE* Eq. 6) over N software projects. The second accuracy criterion is *pred(d)* (see Eq. 8), it represents the percentage of software projects whose MRE is less than or equal to *d*. For this paper, we used *d* =0.25 because it is a commonly used value. The third measure is the median of *MRE* (*MdMRE*) (see Eq. 9).

$$MRE=\left|\frac{cost_{actual}-cost_{predicted}}{cost_{actual}}\right| \tag{6}$$

$$MMRE = {}^{1}/_{Nb}\sum_{i=1}^{Nb} MRE_i \tag{7}$$

$$pred(d) = {}^{k}/_{Nb} \tag{8}$$

$$MdMRE= median(MREi) \tag{9}$$

Where *Nb* represents the number of software projects and *k* is the total number of observations and k is the number of observations whose MRE is less or equal to d.

4.3 Validation method

To generalize the prediction ability of our suggested model, we used a 30% holdout validation technique. This method used 70% of the dataset for learning the model and holdout 30% of available data as a testing set. Unlike using the same dataset for learning and testing the model, this method is advantageous because the model performs prediction on unseen data samples. And therefore produces less biased cost predictions.

5 Results

In this section, we report and discuss all experimental findings of our proposed model. We note that we used different python packages and the open-source scikit-learn library to develop the suggested model. All experiments were performed using a MacBook Pro OS X El Capitan version 10.11.6.

5.1 Results of feature selection

We discuss in this subsection the most finding of the feature selection step. Table 3 lists the removed features and the number of selected ones for each dataset using the Optainet algorithm. While table 4 refers to the results of the study made by Zakrani et.al [8] using random forest (RF) and Boruta feature selection methods.

For the majority of datasets, the rejected features using Optainet are among the least important features identified by the random forest method. Except for Kemerer and Miyazaki where:

For Kemerer: Optainet rejects the AdjFp attribute that is neither rejected by RF nor by the Boruta method.

For Miyazaki: Optainet rejects KLOC, and FORM attributes which are neither ignored by Boruta nor presented among the least important feature for RF.

Also, we note that the Optainet technique almost removed the first unimportant feature identified by RF this is exactly the case for China, Desharnais, ISBSG, Miyazaki, and Kemerer datasets. Nevertheless, for Cocomo and Albrecht, Optainet doesn't remove the first least feature identified by RF. For example, for the Cocomo dataset, Optainet ignores the VIRTmajeur attribute, which is identified as the second unimportant feature by RF, while the first one is VEXP.

We mention that the Optainet method selects almost fifty percent (50%) of attributes for each dataset. Except for the Miyazaki dataset where just one attribute is selected from seven.

Table 3. The number of selected features and removed ones in each dataset using Optainet algorithm.

Datasets	**# of features**	**Optainet**	
		#of selected features	**Removed features**
ISBSG (R8)	6	3	Business, Locations, Concurrent
COCOMO	13	11	VIRTmineur,VIRTmajeur
DESHARNAIS	8	4	TeamExp, ManagerExp, Envergure
ALBRECHT	7	4	Input, File, AdjFP
KEMERER	6	2	Language, Hardware, **AdjFP**, RAWFP
MIYAZAKI	7	1	**KLOC**, SCRN, **FORM**, FILE, ESCRN, EFORM
CHINA	15	12	Changed, Resource, Dev.Type

Table 4. The number of removed features in each dataset using RF and Boruta methods.

Réf.	**Datasets**	**Boruta Removed features**	**Four least important feature RF (1,2,3,4)**
[8]	ISBSG (R8)	Factor; Business;Locations;	Business, Locations, Factor, Concurrent
	COCOMO	-	VEXP, VIRTmajeur, LEXP, VIRTmineur
	DESHARNAIS	TeamExp; ManagerExp	TeamExp, ManagerExp, Language, Envergure
	ALBRECHT	Input	FPAdj, Input, Inquiry, File
	KEMERER	Language, Hardware, Duration	Language, Hardware, Duration, RAWFP
	MIYAZAKI	EFORM	EFORM, ESCRN, FILE, SCRN
	CHINA	Deleted, Dev.Type	Dev.Type, Deleted, Changed, Resource

5.2 Optainet based SVR model results

The Optainet-based SVR model is built using the Optainet algorithm for FS step and also for tuning the hyperparameters of the SVR model. We evaluate the model using Pred(25%), MdMRE, and MMRE measures, while the model validation is made using the 30% holdout technique. We note that we perform two models: 1-the SVR_Optainet_FS_PO which refers to the SVR model using the Optainet method for feature selection (FS) and parameter optimization (PO), 2-the second model is SVR_Optainet_PO which refers to the SVR model using all features and with Optainet for only parameter optimization. These two models were assessed and compared to the models used in [8] which are SVR_BFE, SVR_Boruta, and SVR. Tables 5-7 show the complete experimental evaluations.

We observe from table 5 that SVR_Optainet_FS_PO outperforms all other SVR configurations in terms of Pred (25%). Besides, we mention that SVR_Optainet_PO came second thanks to its best pred in 5 out of 7 datasets. while SVR_Boruta and SVR and SVR_BFE generated acceptable values of Pred exclusively in the china dataset. The highest value of Pred was achieved using SVR_Optainet_FS_PO with 93% in the china dataset followed by SVR-BFE (83.33) in the same dataset.

According to MMRE and MdMRE results (see tables 6-7), we notice that SVR_Optainet_FS_PO also outperforms the majority of other techniques overall used datasets where the minimal values of errors were obtained using China dataset while maximal values are obtained using Cocomo dataset. The only exception is observed in the Albrecht dataset where SVR_Boruta has a lower MdMRE than SVR_Optainet_FS_PO(97% improvement) and for the Cocomo dataset where SVR_Optainet_PO also outperforms SVR_Optainet_FS_PO(37% improvement).

To summarize these results, we can conclude that the SVR model with feature selection combined with parameters optimization improves greatly the performance of the cost predictions. Besides, the Optainet algorithm enhances the cost estimates better than BFE and Boruta Feature selection methods.

6 Conclusion and future work

In this paper, we propose a new SVR model based on the Optainet technique for feature selection and parameters optimization. This model was compared to four other SVR models: the optimized SVR without FS, the SVR model using BFE, the SVR model using Boruta FS, and the SVR without feature selection. The overall evaluation was performed using 70% of data as a training set and 30% holdout for the test set. The results show that our proposed method outperforms all the four SVR configurations based on all dataset in terms of three used criteria: Pred(25%), MdMRE, and MMRE. The SVR_Optainet_FS_PO gave better accuracy than Boruta and BFE feature selection.

Concerning our future work, it would be interesting to generalize the predominance of our proposed model by comparing it to other non-SVR models using LOOCV method.

Table 5. The values of Pred(25%) over all datasets

Methods	**ALBRECHT**	**CHINA**	**COCOMO**	**DESHARNAIS**	**ISBSG**	**KEMERER**	**MIYAZAKI**
SVR_Optainet_FS_PO	**62.5**	**93.3**	**44.7**	**66.6**	**47.8**	**60**	**60**
SVR_Optainet_PO	37.5	55.3	36.8	54.1	43.4	20	40
SVR_BFE	42,857	83,333	36,842	39,13	31,111	20	28,571
SVR_Boruta	28,571	75,333	34,211	34,783	24,444	0	42,857
SVR	42,857	81,333	36.842	30,435	31,111	20	42,857

Table 6. The values of MMRE over all datasets

Methods	**ALBRECHT**	**CHINA**	**COCOMO**	**DESHARNAIS**	**ISBSG**	**KEMERER**	**MIYAZAKI**
SVR_Optainet_FS_PO	1.083	**0.107**	1.312	**0.279**	**0.686**	**0.802**	**0.384**
SVR_Optainet_PO	1.417	0.399	**1.102**	0.305	1.182	1.820	0.409
SVR_BFE	**0,548**	0,17	1,242	0,462	1,092	1,334	1,367
SVR_Boruta	0,566	0,242	1,524	0,457	1,559	1,671	0,527
SVR	0,583	0,187	1.375	0,467	1,478	1,235	0,553

Table 7. The values of MdMRE over all datasets

Methods	**ALBRECHT**	**CHINA**	**COCOMO**	**DESHARNAIS**	**ISBSG**	**KEMERER**	**MIYAZAKI**
SVR_Optainet_FS_PO	**0.235**	**0.040**	0.439	**0.122**	**0.281**	**0.150**	**0.163**
SVR_Optainet_PO	0.378	1.191	**0.320**	0.210	0.346	1.318	0.457

SVR_BFE	0,308	0,047	0,508	0,361	0,429	0,469	0,571
SVR_Boruta	0,343	0,088	0,41	0,332	0,433	0,668	0,311
SVR	0,422	0,054	0.456	0,397	0,526	0,844	0,295

References

1. A. L. I. Oliveira, "Estimation of software project effort with support vector regression," *Neurocomputing*, vol. 69, no. 13–15, pp. 1749–1753, Aug. 2006, doi: 10/fr8fhx.
2. A. Zakrani and A. Idri, "Applying radial basis function neural networks based on fuzzy clustering to estimate web applications effort," *Int. Rev. Comput. Softw.*, vol. 5, pp. 516–524, 2010.
3. A. Najm, A. Zakrani, and A. Marzak, "Decision Trees Based Software Development Effort Estimation: A Systematic Mapping Study," in *2019 International Conference of Computer Science and Renewable Energies (ICCSRE)*, Agadir, Morocco, Jul. 2019, pp. 1–6, doi: 10/gf7785.
4. A. Najm, A. Zakrani, and A. Marzak, "Systematic Review Study of Decision Trees based Software Development Effort Estimation," *Int. J. Adv. Comput. Sci. Appl. IJACSA*, vol. 11, no. 7, Art. no. 7, 34/31 2020, doi: 10/ghfkn3.
5. Q. Liu, J. Xiao, and H. Zhu, "Feature selection for software effort estimation with localized neighborhood mutual information," *Clust. Comput.*, vol. 22, no. 3, pp. 6953–6961, May 2019, doi: 10/gjn3wv.
6. L. N. de Castro and J. Timmis, "An artificial immune network for multimodal function optimization," in *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)*, May 2002, vol. 1, pp. 699–704 vol.1, doi: 10/c7mfqm.
7. J. Yang and V. Honavar, "Feature Subset Selection Using a Genetic Algorithm," in *Feature Extraction, Construction and Selection: A Data Mining Perspective*, H. Liu and H. Motoda, Eds. Boston, MA: Springer US, 1998, pp. 117–136.
8. A. Zakrani, M. Hain, and A. Idri, "Improving Software Development effort estimating using Support Vector Regression and Feature Selection," *IAES Int. J. Artif. Intell. IJ-AI*, vol. 8, no. 4, p. 399, Dec. 2019, doi: 10/gjn3zv.
9. P. L. Braga, A. L. I. Oliveira, and S. R. L. Meira, "A GA-based feature selection and parameters optimization for support vector regression applied to software effort estimation," in *Proceedings of the 2008 ACM symposium on Applied computing*, New York, NY, USA, Mar. 2008, pp. 1788–1792, doi: 10/drgmmp.
10. A. L. I. Oliveira, P. L. Braga, R. M. F. Lima, and M. L. Cornélio, "GA-based method for feature selection and parameters optimization for machine learning regression applied to software effort estimation," *Inf. Softw. Technol.*, vol. 52, no. 11, pp. 1155–1166, Nov. 2010, doi: 10/dw8fh3.
11. D. Novitasari, I. Cholissodin, and W. F. Mahmudy, "Hybridizing PSO With SA for Optimizing SVR Applied to Software Effort Estimation," *TELKOMNIKA Telecommun. Comput. Electron. Control*, vol. 14, no. 1, Art. no. 1, Mar. 2016, doi: 10/gjn3z3.
12. A. Corazza, S. Di Martino, F. Ferrucci, C. Gravino, and E. Mendes, "Investigating the use of Support Vector Regression for web effort estimation," *Empir. Softw. Eng.*, vol. 16, no. 2, pp. 211–243, Apr. 2011, doi: 10/ch8s8k.

13. A. Corazza, S. Di Martino, F. Ferrucci, C. Gravino, and E. Mendes, "Using Support Vector Regression for Web Development Effort Estimation," in *Software Process and Product Measurement*, Berlin, Heidelberg, 2009, pp. 255–271, doi: 10/d4rhss.
14. A. Corazza, S. D. Martino, F. Ferrucci, C. Gravino, and E. Mendes, "Applying support vector regression for web effort estimation using a cross-company dataset," in *2009 3rd International Symposium on Empirical Software Engineering and Measurement*, Oct. 2009, pp. 191–202, doi: 10/frkkms.
15. A. Corazza, S. Di Martino, F. Ferrucci, C. Gravino, F. Sarro, and E. Mendes, "Using tabu search to configure support vector regression for effort estimation," *Empir. Softw. Eng.*, vol. 18, no. 3, pp. 506–546, Jun. 2013, doi: 10/fnq2hj.
16. V. Vapnik, *The Nature of Statistical Learning Theory*, 2nd ed. New York: Springer-Verlag, 2000.
17. A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Stat. Comput.*, vol. 14, no. 3, pp. 199–222, Aug. 2004, doi: 10/fsv7qx.
18. C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sep. 1995, doi: 10/cv7fn6.
19. M. Hosni, A. Idri, A. Abran, and A. B. Nassif, "On the value of parameter tuning in heterogeneous ensembles effort estimation," *Soft Comput.*, vol. 22, no. 18, pp. 5977–6010, Sep. 2018, doi: 10/gd6rhh.
20. A. Idri and I. Abnane, "Fuzzy Analogy Based Effort Estimation: An Empirical Comparative Study," in *2017 IEEE International Conference on Computer and Information Technology (CIT)*, Helsinki, Finland, Aug. 2017, pp. 114–121, doi: 10/ghp8xv.
21. L. N. de Castro and F. J. V. Zuben, "aiNet: An Artificial Immune Network for Data Analysis," p. 40, doi: 10/fdxcbf.
22. N. K. Jerne, "Towards a network theory of the immune system," *Ann. Immuno 125C*, pp. 373–389, 1974.
23. V. Cutello, G. Nicosia, and M. Pavone, "Exploring the Capability of Immune Algorithms: A Characterization of Hypermutation Operators," in *Artificial Immune Systems*, Berlin, Heidelberg, 2004, pp. 263–276, doi: 10/fnqf2t.
24. J. Kelsey and J. Timmis, "Immune Inspired Somatic Contiguous Hypermutation for Function Optimisation," in *Genetic and Evolutionary Computation — GECCO 2003*, Berlin, Heidelberg, 2003, pp. 207–218, doi: 10/dhfb4r.
25. L. De Castro and F. Von Zuben, "The Clonal Selection Algorithm with Engineering Applications," *Artif. Immune Syst.*, vol. 8, Feb. 2001.
26. C. J. Burgess and M. Lefley, "Can genetic programming improve software effort estimation? A comparative evaluation," *Inf. Softw. Technol.*, vol. 43, no. 14, pp. 863–873, Dec. 2001, doi: 10/bk4w52.
27. P. L. Braga, A. L. I. Oliveira, G. H. T. Ribeiro, and S. R. L. Meira, "Bagging Predictors for Estimation of Software Project Effort," in *2007 International Joint Conference on Neural Networks*, Aug. 2007, pp. 1595–1600, doi: 10.1109/IJCNN.2007.4371196.
28. Miyoung Shin and A. L. Goel, "Empirical data modeling in software engineering using radial basis functions," *IEEE Trans. Softw. Eng.*, vol. 26, no. 6, pp. 567–576, Jun. 2000, doi: 10/dxsjxc.