Fuzzy Systems and Data Mining VI
A.J. Tallón-Ballesteros (Ed.)

doi:10.3233/FAIA200730

The Mechanism Analysis of the Accelerator for Support Vector Regression Based on Data Partition

Yunsheng SONG [a,1], Fangyi LI [a], Jianyu LIU [a] and Juao ZHANG [a]
[a] *School of Information Science and Engineering, Shandong Agricultural University, Tai'an, 271018, Shandong, China*

Abstract. Support vector regression is an important algorithm in machine learning, and it is widely used in real life for its good performance, such as house price forecast, disease prediction, weather forecast, and so on. However, it cannot efficiently process large-scale data, because it has a high time complexity in the training process. Data partition as an important solution to solve the large-scale learning problem mainly focuses on the classification task, it trains the classifiers over the divided subsets produced by data partition and obtain the final classifier by combining those classifiers. Meanwhile, the most existing method rarely study the influence of data partition on the regressor performance, so that it is difficult to keep its generation ability. To solve this problem, we obtain the estimation of the difference in objective function before and after the data partition. Mini-Batch K-Means clustering is adopted to largely reduce this difference, and an improved algorithm is proposed. This proposed algorithm includes training stage and prediction stage. In training stag, it uses Mini-Batch K-Means clustering to divide the input space into some disjoint sub-regions of equal sample size, then it trains the regressor on each divided sub-region using support vector regression algorithm. In the prediction stage, the regressor merely offers the predicted label for the unlabeled instances that are in the same sub-region. Experiment results on real datasets illustrate that the proposed algorithm obtains the similar generation ability as the original algorithm, but it has less execution time than other acceleration algorithms.

Keywords. Machine learning, Large-scale data, Data partition, Support vector regression, Generation ability

1. Introduction

As the volume of data becomes increasingly large, it takes a revolutionary change for training support vector regression (SVR) model for its time complexity and space complexity. Lots of improved algorithms are proposed to accelerate the training process and minimize losses on the generalization performance. The proposed algorithms are classified into reduction algorithms and decomposition algorithms [1]. The main style of reduction algorithms is that the training samples nearby the hyperplane have a large con-

[1]Corresponding Author: Yunsheng Song, Shandong Agricultural University, Tai'an, Shandong, China; E-mail: sys_sd@126.com

tribution to SVR model than others [2–6]. In this way, these proposed algorithms seeks the critical samples to form a new subset instead of the training set, where the size of the selected subset tends to be smaller than the original training set. Therefore, the regressor trained over the selected subset has a high training efficiency. In fact, the reduction algorithms want to find the support vectors to keep the prediction performance. However, the support vectors are determined by the regressor, and it is difficult to obtain before training the model. Meanwhile, the reduction algorithms usually requires to read the entire training data several times, so that they needs lots of time to get the final result [7–10]. Decomposition algorithms train the SVR model with all the training instances rather than some of them. The existing decomposition algorithms focus on how to obtain the approximate performance as the original SVR using optimization strategy, while they still deal with all the data once or more [11–13]. Therefore, these algorithms could not handle the big data well.

2. Related Concept

Let $T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$ mark the training set, where each instance x_i is expressed with m attributes, $y_i \in R$ is its output, and N is the number of labeled instances. The main task in training a SVR model over T is to solve the following quadratic optimization problem:

$$\min_{\theta,\rho \in R^N} \psi(\theta,\rho) \;\; s.t.\; 0 \le \theta, \rho \le C, e^T(\theta - \rho) = 0 \tag{1}$$

where the objective function $\psi(\theta,\rho) = \{\frac{1}{2}(\theta-\rho)^T \mathbb{K}(\theta-\rho) + \varepsilon e^T(\theta+\rho) - Y^T(\theta-\rho)\}$, $\theta = (\theta_1, \theta_2, \cdots, \theta_N)^T \in R^N$ and $\rho = (\rho_1, \rho_2, \cdots, \rho_N)^T \in R^N$ are the vectors of dual variables; $\mathbb{K}$ is an $N \times N$ matrix with $\mathbb{K}_{ij} = K(x_i, x_j)$, and $K(x_i, x_j)$ is the kernel function; C is the balancing parameter between loss and regularization in the SVR primal problem; e_N is the vector of all ones, andε is the relaxation factor; $Y = (y_1, y_2, \cdots, y_N)$ is the output vector of the training set. Let $\theta^* = (\theta_1^*, \theta_2^*, \cdots, \theta_N^*)$ and $\rho^* = (\rho_1^*, \rho_2^*, \cdots, \rho_N^*)$ denote the optimal solution of the problem (1), the predicted value for a test data x can be computed by the decision function $g(x) = \sum_{i=1}^{N} (\theta_i^* - \rho_i^*) K(x, x_i) + b$, where the bias term $b = 0$, because any improvement in test accuracy is not observed by it in this paper.

The main idea behind accelerating the process of training SVR based on data partition is to divide the training set into smaller disjoint subsets, and then each model can be trained independently and efficiently over each subset. Suppose the training dataset T can be decomposed into n sets $T_l (l = 1, 2, \cdots, n)$, $\bigcup_{l=1}^{n} = T$ and $T_l \bigcap_{l \ne k} T_k = \emptyset$, where n is prefixed by the user. Then each subproblem over each divided subset T_l is that

$$\min_{\theta_{(l)},\rho_{(l)}} \psi_l(\theta_{(l)}, \rho_{(l)}) \;\; s.t.\; 0 \le \theta_{(l)}, \rho_{(l)} \le C, e_{(l)}^T(\theta_{(l)} - \rho_{(l)}) = 0 \tag{2}$$

where $\psi_l(\theta_{(l)},\rho_{(l)}) = \{\frac{1}{2}(\theta_{(l)}-\rho_{(l)})^T\mathbb{K}_{l\times l}(\theta_{(l)}-\rho_{(l)}) + \varepsilon e^T(\theta_{(l)}+\rho_{(l)}) - Y_{(l)}^T(\theta_{(l)}-\rho_{(l)})\}$, $\theta_{(l)},\rho_{(l)}$ denote the subvector $\{\theta_i|(x_i,y_i)\in T_l\}$ and the subvector $\{\rho_i|(x_i,y_i)\in T_l\}$ respectively, $\mathbb{K}_{l\times l}$ is the submatrix of $\mathbb{K}$ with row and column indexes T_l, and $Y_{(l)}$ is the output vector of the instances over the subset T_l, $0\le\theta_{(l)},\rho_{(l)}\le C, e^T(\theta_{(l)}-\rho_{(l)})=0$. I In this way, the optimal problem (1) is approximately divided into n subproblems. Then the optimal solutions of all the subproblems are combined to initialize a coordinate descent solver for the original problem. Let $\widehat{\theta}_{(l)},\widehat{\rho}_{(l)}$ are the optimal solution of the subproblem $\min\limits_{\theta_{(l)},\rho_{(l)}\in R^N} f_l(\theta_{(l)},\rho_{(l)})$. Then we could concatenate all the subproblem solutions to form an approximate solution $\widehat{\theta}$ and $\widehat{\rho}$ for the whole problem, where $\widehat{\theta}=[\widehat{\theta}_{(1)},\widehat{\theta}_{(2)},\widehat{\theta}_{(3)},\cdots,\widehat{\theta}_{(n)}]$ and $\widehat{\rho}=[\widehat{\rho}_{(1)},\widehat{\rho}_{(2)},\widehat{\rho}_{(3)},\cdots,\widehat{\rho}_{(n)}]$.

3. MK-SVR algorithm

In this section, we discuss the relationship between the raw problem (1) and the divided subproblems (2). A kernel function $\widehat{K}(x_i,x_j)=I_{\iota(x_i),\nu(x_j)}K(x_i,x_j)$ is defined, $\widehat{K}(x_i,x_j)=I_{\iota(x_i),\nu(x_j)}K(x_i,x_j)$, where $\iota(x_i)$ is the the divided subset that the instance x_i belongs to; $I_{\iota(x_i),\iota(x_j)}=1$ iff $\iota(x_i)=\iota(x_j)$, otherwise $I_{\iota(x_i),\nu(x_j)}=0$.

Lemma 1. *The vector pair $[\widehat{\theta},\widehat{\rho}]$ could be the solution of the optimization problem (1) with kernel function $K(.,.)$ replaced by $\widehat{K}(.,.)$, where the objective function of this optimization problem is $\widehat{\psi}(\theta,\rho)=\{\frac{1}{2}(\theta-\rho)^T\widehat{\mathbb{K}}(\theta-\rho)+\varepsilon e^T(\theta+\rho)-Y^T(\theta-\rho)\}$, $\widehat{\mathbb{K}}$ is a $n\times n$ kernel matrix with each element $\widehat{\mathbb{K}}_{ij}=\widehat{K}(x_i,x_j)$.*

Proof. According to its definition of $\widehat{K}(x.,.)$, $\widehat{K}(x_i,x_j)=0$ if $\iota(x_i)\neq\iota(x_j)$, and $\widehat{K}(x_i,x_j)=K(x_i,x_j)$ if $\iota(x_i)=\iota(x_j)$. Then the function $\widehat{f}(\theta,\rho)$ can be decomposed into the sum of several functions.

$$\begin{aligned}
\widehat{\psi}(\theta,\rho) &= \tfrac{1}{2}(\theta-\rho)^T\widehat{\mathbb{K}}(\theta-\rho)+\varepsilon e^T(\theta+\rho)-Y^T(\theta-\rho)\\
&= \tfrac{1}{2}\sum_{\iota(x_i)=\iota(x_j)}(\theta_i-\rho_i)\widehat{K}(x_i,x_j)(\theta_j-\rho_j)+\tfrac{1}{2}\sum_{\iota(x_i)\neq\iota(x_j)}(\theta_i-\rho_i)\widehat{K}(x_i,x_j)(\theta_j-\rho_j)\\
&+ \sum_{i=1}^{N}\{\varepsilon(\theta_i+\rho_i)-y_i(\theta_i-\rho_i)\}\\
&= \tfrac{1}{2}\sum_{\iota(x_i)=\iota(x_j)}(\theta_i-\rho_i)\widehat{K}(x_i,x_j)(\theta_j-\rho_j)+\sum_{i=1}^{N}\{\varepsilon(\theta_i+\rho_i)-y_i(\theta_i-\rho_i)\}\\
&= \sum_{l=1}^{n}\{\tfrac{1}{2}(\theta_{(l)}-\rho_{(l)})^T\mathbb{K}_{l\times l}(\theta_{(l)}-\rho_{(l)})+\varepsilon e^T(\theta_{(l)}+\rho_{(l)})-Y_{(l)}^T(\theta_{(l)}-\rho_{(l)})\}\\
&= \sum_{l=1}^{n}\widehat{f}_l(\theta_l,\rho_l).
\end{aligned} \tag{3}$$

From Eq. (3), the problem $\min\limits_{\theta,\rho\in R^N}\widehat{f}(\theta,\rho)$ is divided into n subproblems (2) and these subproblems are independent. As the subvector pair $[\widehat{\theta}_{(l)},\widehat{\rho}_{(l)}]$ could be the solution of the optimal subproblem ($l=1,2,\cdots,n$), the vector pair $[\widehat{\theta},\widehat{\rho}]$ must be the solution of the original problem $\min\limits_{\theta,\rho}\widehat{f}(\theta,\rho)$. Furthermore, $e^T(\widehat{\theta}-\widehat{\rho})=\sum\limits_{l=1}^{n}e_{(l)}^T(\widehat{\theta}_{(l)}-\widehat{\rho}_{(l)})=0$,

each element h of the vector $\widehat{\theta}_{(l)}$ satisfies the condition $0 \le h \le C$, and each element of the vector $\widehat{\rho}_{(l)}$ also satisfies this condition. Therefore, the vector pair $[\widehat{\theta},\widehat{\rho}]$ satisfies the restrictions of the problem (1). □

From the **Lemma 1**, we can find that the accelerating SVR algorithm using the divide-and-conquer strategy is equivalent to solve the problem $\min_{\theta,\rho\in R^N} \widehat{f}(\theta,\rho)$. Now we discuss the difference between $f(\widehat{\theta},\widehat{\rho})$ and $f(\theta^*,\rho^*)$, so as to study the effect of data partition to the generation ability of this algorithm.

Theorem 1.

$$0 \le \psi(\widehat{\theta},\widehat{\rho}) - \psi(\theta^*,\beta^*) \le (1/2)C^2 D(v)$$

where $D(\tau) = \sum_{i,j:\iota(x_i)\neq\iota(x_j)} |K(x_i,x_j)|$.

Proof. Based on the mathematical notation, we can get

$$\begin{aligned}\widehat{\psi}(\theta^*,\beta^*) &= \tfrac{1}{2}(\theta^*-\beta^*)^T\widehat{\mathbb{K}}(\theta^*-\beta^*) + \varepsilon e^T(\theta^*+\beta^*) - Y^T(\theta^*-\beta^*) \\ &= \tfrac{1}{2}\sum_{i,j:\iota(x_i)=\iota(x_j)} (\alpha_i^*-\beta_i^*)K(x_i,x_j)(\alpha_j^*-\beta_j^*) + \varepsilon e^T(\alpha^*+\beta^*) - Y^T(\alpha^*-\beta^*) \\ &= \psi(\alpha^*,\beta^*) - \tfrac{1}{2}\sum_{i,j:\iota(x_i)\neq\iota(x_j)} (\alpha_i^*-\beta_i^*)K(x_i,x_j)(\alpha_j^*-\beta_j^*)\end{aligned} \tag{4}$$

$$\widehat{\psi}(\widehat{\theta},\widehat{\rho}) = \psi(\widehat{\theta},\widehat{\rho}) - \frac{1}{2}\sum_{i,j:\iota(x_i)\neq\iota(x_j)} (\widehat{\theta}_i-\widehat{\rho}_i)K(x_i,x_j)(\widehat{\theta}_j-\widehat{\rho}_j) \tag{5}$$

Furthermore, $\widehat{\psi}(\widehat{\theta},\widehat{\rho}) \le \widehat{\psi}(\theta^*,\rho^*)$, because the vector pair $[\widehat{\theta},\widehat{\rho}]$ is the optimal solution of the problem (2). Combining with Eqs.(4) and (5), and $0 \le \theta_i^*,\rho_i^*,\widehat{\theta}_i,\widehat{\rho}_i \le C$ for all i, then

$$\begin{aligned}\psi(\widehat{\theta},\widehat{\rho}) &= \widehat{\psi}(\widehat{\theta},\widehat{\rho}) + \tfrac{1}{2}\sum_{i,j:\iota(x_i)\neq\iota(x_j)} (\widehat{\theta}_i-\widehat{\rho}_i)K(x_i,x_j)(\widehat{\theta}_j-\widehat{\rho}_j) \\ &\le \widehat{\psi}(\theta^*,\rho^*) + \tfrac{1}{2}\sum_{i,j:\iota(x_i)\neq\iota(x_j)} (\widehat{\theta}_i-\widehat{\rho}_i)K(x_i,x_j)(\widehat{\theta}_j-\widehat{\rho}_j) \\ &= \psi(\theta^*,\rho^*) + \tfrac{1}{2}\sum_{i,j:\iota(x_i)\neq\iota(x_j)} [(\widehat{\theta}_i-\widehat{\rho}_i)-(\theta_i^*-\rho_i^*)]K(x_i,x_j)[(\widehat{\theta}_j-\widehat{\rho}_j)-(\theta_j^*-\rho_j^*)] \\ &\le \psi(\theta^*,\rho^*) + \tfrac{1}{2}C^2 D(\iota).\end{aligned} \tag{6}$$

□

In order to minimize the difference between $\psi(\widehat{\theta},\widehat{\rho})$ and $\psi(\theta^*,\rho^*)$, we need a data partition algorithm with small $D(v)$. This kernel function is one kind of similarity measure according to its definition. When the kernel function is the radical basis function , the value of $K(x_i,x_j)$ is proportional to the Euclidean distance of the instances x_i and x_j. In this way, $D(\iota) = \sum_{i,j:\iota(x_i)\neq\iota(x_j)} K(x_i,x_j)$, and it is the sum of the similarity between

instances in different divided subsets. Min-Batch K-Means clustering algorithm that is one is one kind of improved algorithm can obtain small $D(v)$ using Euclidean distance. It produces the subset subsets of approximately equal size for large-scale data, because it inherits the uniformity effect of K-Means clustering algorithm.

As the dataset T can be decomposed into n disjoint divided sets $T_i(\bigcup_{l=1}^{n} = T, T_i \bigcap_{i \neq j} T_j = \emptyset)$ using Min-Batch K-Means clustering algorithm, and the input space χ is also divided into n disjoint region χ_l, where $l = 1, 2, \cdots, n$. The SVR regressor $\widehat{g}_l(x)$ trained over the divided subset T_l only uses the information of the divided region χ_l, so it could effectively predict the instances within the the divided region χ_l rather than the instances not in other regions. In this way, the final regressor $\widehat{g}(x)$ is that

$$\widehat{g}(x) = \sum_{i=1}^{l} I_{x,\chi_l} \widehat{g}_l(x) \tag{7}$$

where $I_{x,\chi_l} = 1$ if and only if $x \in \chi_l$ and $I_{x,\chi_l} = 0$ otherwise, and the region which the instance x belongs to can be computed by finding the nearest cluster center. Therefore, the predicting process firstly searches the nearest cluster which the given unlabeled instance x_0 belongs to, and uses the regressor obtained by data within that cluster to compute its predicted value.

3.1. Time complexion analysis

The time complexion is very important for the algorithms for large-scale data. The proposed algorithm includes the data partition and training the regressors on the divided subsets. The first process has a time complexion of $O(N)$, and the time complexion of the second process is $O(s^3)$, where n and s are the training instances number and the size of the maximum divided subsets. Then the time complexion of our proposed algorithm is $O(N + s^3)$. So the size of divided subsets is inversely proportional to the time complexion. On the other hand, the difference $D(\tau)$ could be small as the large size of divided subsets, the proposed algorithm with the largely divided subsets has a better generation ability than one with the small divided subsets.

4. Experiment

4.1. Data Information and Related Setup

Nine representative real datasets are chosen for their characteristic from the LIBSVM datasets [14] and UCI datasets [15]. The information of the selected datasets is shown in Table 1, where the size of each dataset is larger than 10000.

Table 1 Information of nine datasets

Datasets	Size	Features
ailerons	13750	40
cada	20640	8
california	20640	8
house	22784	16
mv	40768	10
pole	14998	26
Aggregates	200000	4
sgemm	241600	10
YearMSD	515345	90

Two typical algorithms for accelerating SVR algorithm are chosen in this paper: LIBSVM and SVR based on random data partition (RP-SVR). Mean square error (MSE), coefficient of determination (R^2), and execution time (ET) in seconds are chosen for evaluating their performance, and the first two indicators evaluate the prediction ability [16]. All the measurement results is computed using 10-fold cross-validation method, and $n = N^{0.3}$ [17].

4.2. Prediction Ability

Table 2 shows the prediction ability evaluated by MSE and R^2 of three algorithms.

Table 2 Prediction performance on nine datasets

Datasets	MK-SVR		RP-SVR		LIBSVM	
	MSE	R^2	MSE	R^2	MSE	R^2
ailerons	0.735	0.003	0.685	0.004	0.821	0.002
cadata	0.664	0.02	0.609	0.023	0.663	0.02
california	0.367	8.81E+09	-0.066	1.48E+10	-0.064	1.48E+10
house	0.07	2.83E+09	-0.105	3.36E+09	-0.105	3.36E+09
mv	0.977	2.362	0.774	23.417	0.989	1.112
pole	0.415	1.05E+03	-0.478	2.66E+03	-0.343	2.41E+03
Aggregates	0.767	1.68E+04	0.437	40514.548	0.778	1.60E+04
sgemm	0.182	1.10E+05	0.022	1.31E+05	0.364	8.55E+04
YearMSD	0.214	114.417	0.1826	105.56	0.22	117.625
Mean	0.488	1.29E+09	0.229	2.02E+09	0.369	2.02E+09
Median	0.415	1.05E+03	0.1826	2.66E+03	0.364	2.41E+03

On the whole, the average values of MSE for three algorithms are 1.29E+09, 2.02E+09 and 2.02E+09, as well as the medians of MSE are 1.05E+03, 2.66E+03 and 2.41E+03 from **Table 2**. So it indicts that the MK-SVR algorithm obtains a similar prediction accuracy as LIBSVM algorithm, but better than the RP-SVR algorithm. This significant result is shown on datasets california, house and pole. For another prediction performance measurement R^2, the average values of R^2 for three algorithms are 0.488, 0.229, 0.369, as well as the medians of R^2 are 0.415, 0.183, and 0.364. In conclusion, MK-SVR algorithm has a better prediction performance than the RP-SVR algorithm, similar to LIBSVM algorithm.

4.3. The execution time

Table 3 lists the execution time (in seconds) of these three algorithms. ET of MK-SVR is much smaller than that of the other two algorithms on most of the datasets .

Table 3 ET of three algorithms on nine datasets

Dataset	MK-SVR	RP-SVR	LIBSVM
ailerons	0.086	0.123	0.416
cadata	0.39	0.563	4.827
california	0.888	1.306	13.232
house	1.446	2.149	20
mv	1.495	4.852	40.556
pole	1.053	1.663	38.122
Aggregates	30.869	84.437	2365.737
sgemm	57.184	146.269	4396.376
YearMSD	138.547	156.325	48714.719
Mean	231.958	397.687	55594.167
Median	1.446	2.149	38.122

The average values of ET for three algorithms are 231.958, 397.687 and 55594.167, as well as the medians of ET are 1.446, 2.149, 38.122. Then MK-SVR has less training time both than RP-SVR algorithm and LIBSVM algorithm. Meanwhile, the training time of MK-SVR algorithm is also less than other algorithms on each dataset from **Table 3**. Compared with other algorithms, MK-SVR algorithm is very efficient in processing large-scale data. The reason for this issue is that they have different training process and data partition strategy.

5. Conclusion

This paper analyzes the influence of data partition on the performance of SVR model, and finds that Mini-Batch K-Means clustering could largely reduce this effect. Therefore, we have given an improved SVR algorithm using the Mini-Batch K-Means clustering. In the training process, the input space spanned by the training instances is divided into several disjoint regions, and each SVR model is trained using the instances within each divided region independently. In the prediction process, the SVR model over the divided region gives the predicted outputs of the unlabeled instances within that region. Analysis and experiment results show that our algorithm can obtain a good performance as the original SVR algorithm, but better than another representative algorithm. In the future, we will consider how to adaptively determinate the number of divided subsets for different datasets.

Acknowledgements

This work is supported by the Major Scientific and Technological Innovation Project of Shandong Province (No.2019JZZY01071).

References

[1] Song Y, Liang J, Wang F. An accelerator for support vector machines based on the local geometrical information and data partition. International Journal of Machine Learning and Cybernetics. 2018;10(9):2389-2400.

[2] Cachin C. Pedagogical pattern selection strategies. Neural Networks. 1994;7(1):175-181.

[3] Foody G. The significance of border training patterns in classification by a feedforward neural network using back propagation learning. International Journal of Remote Sensing. 1999;20(18):3549-3562.

[4] Zhang, C., Li, D. and Liang, J., 2020. Multi-granularity three-way decisions with adjustable hesitant fuzzy linguistic multigranulation decision-theoretic rough sets over two universes. Information Sciences, 507, pp.665-683.

[5] Adnan, R., Liang, Z., Heddam, S., Zounemat-Kermani, M., Kisi, O. and Li, B., 2020. Least square support vector machine and multivariate adaptive regression splines for streamflow prediction in mountainous basin using hydro-meteorological data as inputs. Journal of Hydrology, 586, p.124371.

[6] Yang, L. and Dong, H., 2019. Robust support vector machine with generalized quantile loss for classification and regression. Applied Soft Computing, 81, p.105483.

[7] Han X, Clemmensen L. On Weighted Support Vector Regression. Quality and Reliability Engineering International. 2014;30(6):891-903.

[8] Guo G, Zhang J. Reducing examples to accelerate support vector regression. Pattern Recognition Letters. 2007;28(16):2173-2183.

[9] Pan X, Yang Z, Xu Y, Wang L. Safe screening rules for accelerating twin support vector machine classification. IEEE Transactions on Neural Networks and Learning Systems. 2018; 29(5):1876-1887.

[10] Pan X, Pang X, Wang H, Xu Y. A safe screening based framework for support vector regression. Neurocomputing. 2018; 287:163-172.

[11] Ho C, Lin C. Large-scale linear support vector regression. Journal of Machine Learning Research. 2012; 13(1): 3323-3348.

[12] JimenezAlex S, Aviles A, Galn L, Flores A, Matovelle C, Vintimilla, C. Support vector regression to downscaling climate big data: an application for precipitation and temperature future projection assessment. In: Conference on Information Technologies and Communication of Ecuador. 2019 Nov 27-29 ; Cuenca, Ecuador: Springer Press, c2019. p. 182-193.

[13] Wang K, Pei H, Ding X, Zhong P. Robust Proximal Support Vector Regression Based on Maximum Correntropy Criterion. Scientific Programming. 2019;2019:1-11.

[14] Chang CC, Lin CJ. Libsvm: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology. 2011; 2(3):27-66.

[15] UCI Machine Learning Repository [Internet]. Archive.ics.uci.edu. 2020 [cited 13 August 2020]. Available from: http://archive.ics.uci.edu/ml/index.php

[16] Han J, Kamber M, Pei J. Data mining. Burlington: Elsevier Science; 2012.

[17] Kleiner A, Talwalkar A, Sarkar P, Jordan MI.A scalable bootstrap for massive data. Journal of the Royal Statistical Society: Series B (Statistical Methodology). 2014; 76(4):795–816