2020 IEEE 2nd International Conference on Electronics, Control, Optimization and Computer Science (ICECOCS) | 978-1-7281-6921-7/20/$31.00 ©2020 IEEE | DOI: 10.1109/ICECOCS50124.2020.9314373

Optimization of Machine Learning Algorithms Hyper-Parameters for Improving the Prediction of Patients Infected with COVID-19

Soufiane HAMIDA(1), **Oussama EL GANNOUR**(1), **Bouchaib CHERRADI**(1),(2), **Hassan OUAJJI**(1), **Abdelhadi RAIHANI**(1)

(1) SSDIA Laboratory, ENSET of Mohammedia, Hassan II University of Casablanca, 28820, Mohammedia, Morocco.
(2) STIE Team, CRMEF Casablanca-Settat, provincial section of El Jadida, 24000, El Jadida, Morocco.

{hamida.93s, oussama.elgannour, bouchaib.cherradi}@gmail.com, ouajji@hotmail.com, abraihani@yahoo.fr

***Abstract*— In the modeling domain, the selection of appropriate hyper-parameters for classification or prediction algorithms is a difficult task, which has an impact on generalization capacity and classifier performance. In this paper, we compared the performance of five Machine Learning (ML) algorithms from different categories namely: SVM, AdaBoost, Random Forest, XGBoost and Decision Tree. In the first experiment, we adopt a default setting of each model for training and testing. In the second experiment, we use the GridSearch function to find an optimal configuration of the model. The experiments are performed on dataset of anonymous patients with or without COVID-19 disease. The used dataset is obtained from the Albert Einstein Hospital in Sao Paulo, Brazil. To evaluate the reached results, we used different performance evaluation metrics such as: accuracy, precision, recall, AUC and F1-score. The results of the proposed approach have shown that the optimization of the hyper-parameters of the studied learning models leads to an improvement of 18% in terms of Recall.**

***Index Terms*— optimization, hyper-parameters, coronavirus, COVID-19, machine learning, model.**

I. Introduction

Distributed reports on coronavirus disease (COVID-19) change daily and even hourly. A public health emergency has already swept the world: the virus threatens people's lives, affects business and interferes with travel. On January 30, the World Health Organization (WHO) declared the outbreak of a new coronavirus as a public health emergency of international scope and, on March 11, a world pandemic. The gravity of the situation requires seeking a solution to this health problem in all possible ways. The presence of the virus is detected using virologic tests or PCR tests. They consist of a nasopharyngeal swab: a kind of large cotton swab is inserted into the patient's nose in order to collect a certain amount of virus. The sample is sent for analysis to a laboratory. This type of test requires hours of analysis to obtain a result.

Unfortunately, COVID-19 testing capacity is still weak in many countries around the world. Therefore, we need other tools to early diagnose patients in order to stop the spread of the virus. An early diagnosis helps to understand the spread of the virus and takes evidence-based action to slow the spread of the pandemic. In February 2020, a WHO report that Artificial Intelligence (AI) and big data analysis can be a key solution to the problem of the COVID-19 pandemic.

Machine Learning (ML) [1] as branch of AI is a scientific discipline concerned by the development, analysis and implementation of automatable methods that allow a machine to evolve through a learning process. With the rapid development of computer technology, ML has been widely applied in the medical field [2]–[4].

In the medical field, databases are often of large size; high performance computing (HPC) oriented graphics processors (GPU) have been used a lot in ML applications to reduce runtime [5]–[8]. In addition, ML and Deep Learning (DL) proven invaluable in predicting positive cases of multiple diseases [9],[10]. A ML model is defined as a mathematical model with a number of parameters that must be tuned from the data [11]. However, certain parameters called hyper-parameters cannot be studied directly. They are usually chosen by a user based on a certain intuition and test before the start of learning. These parameters demonstrate their importance in improving the performance of the model, such as its complexity or its learning speed. Models can have many hyper-parameters. In fact, finding the best parameters combination can be considered as a separate research problem.

The present study aimed to improve the performance of ML classification models by optimizing the used algorithms parameters. In addition, we aim to establish an early detection model to distinguish COVID-19 cases from healthy ones using viral and blood patient's data with ML techniques. Thus, the main purpose of this paper is to compare the classification/prediction algorithms results using the *GridSearch* function. This function describes the parameters that can be tested on the model for its training process. A parameter grid is defined as a dictionary, where the keys are parameters names and the values consist of a range of values that need to be checked [12]. For this end, we choose five ML algorithms from different categories. These algorithms are trained and tested using a COVID-19 dataset containing the results of blood and viral tests.

The rest of this paper is structured as follows: Section II presents a brief overview of some relevant related work. Section III describes the used dataset structure and the different studied machine learning algorithms. In Section IV, we describe the elements and the stages of the proposed prediction methodology. Section V presents the experimental results and discuss the performance evaluation measures. Finally, section VI concludes this paper and presents some perspectives at this work.

II. Literature Survey

In recent decades, a wide variety of studies has proposed systems for predicting chronic or cancerous diseases. The aim of these studies is to improve the performance and obtain a powerful prediction model. The parameters selection is one of the key problems in modeling because these parameters have a significant impact on the precision performance metric of the model forecast [13]. In programming field, there are many functions which exhaustively search for hyperparameter values. These functions are generally carried out on a computation cluster because they are expensive in terms of computation. The simplest and most used one is the *GridSearch* function [14].

In [7], the authors present an integrated view of the methods used in hyperparametric optimization of learning systems. it emphasizes the hyperparametric optimization aspects by using a combination of techniques for: optimization, space research and reduction of training time.

Another study published in [15], proposed an approach that aims to accelerate the grid search function in order to select the optimal parameters for SVM classifiers. The main idea of this approach was to avoid starting again from zero the training of each SVM model by using the support vectors obtained during the training process of an SVM with a smaller value of C parameter. As training models for an SVM with a higher value rather than using the drive assembly.

In [16], the authors proposed a clinical decision-making system that can helps the doctor to diagnose patients influenced by Parkinson disease. In this study, they invested in the optimization of the *GridSearch* function. Then, develop an optimized deep learning model to predict the early onset of Parkinson disease. Fine-tuning the parameters of the deep learning model provides an overall test accuracy of 89.23% and an average classification accuracy of 91.69%.

Another research work [17] applied the optimization of hyper-parameter of Bayesian algorithm on the CIFAR-10 dataset to improve model performance. indeed, Bayesian optimization clearly obtains optimized values for all the hyper-parameters. This saves time and improves performance. The results show that the error was reduced by 6.2% on the graphics processing unit during validation.

The present work aims to propose a new prediction model optimized for the diagnosis of patients with COVID-19 based on the results of blood samples. The proposed approach is based on two experiments: the first consists of training and testing five machine learning models with their default configuration. In the second experiment, we use *GridSearch* function to determine the best configuration of the models.

III. Materials and Methods

In this section, we provide a description of the dataset and the used classifier methods in the experiments.

A. Dataset Collection

The used database contains anonymized data from patients seen at Albert Einstein Hospital (AEH). This dataset contains two classes: healthy patients (Negative cases) and patients infected with COVID19 (Positive cases). In the other hand, this dataset has 5644 instances and 111 features but with several missing values. For that, it was necessary to carry out a preprocessing exploration by removing the null values. Then, we eliminate features that have a data gap that exceed 80%.

B. Machine learning algorithms

In this sub-section, we gave a brief overview of the five used ML algorithms.

1) Support vector machine

It is a complete set of algorithms needed to solve classification analysis and regression problems. SVM is based on the position of an object in an N-dimensional space belongs to one of the two classes. The support vector method constructs a hyperplane of dimension *N-1* so that all the objects appear in one of the two groups [18]. In Figure 1, this can be represented as follows: there are instances of two different classes (Positive and Negative), and they can be divided linearly.

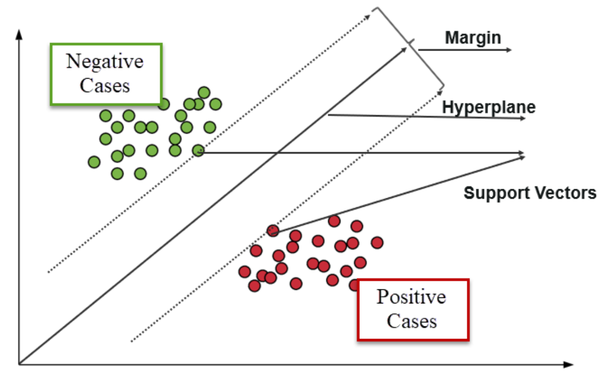

Fig. 1. Diagram representing the SVM classification of binary class data (Positive, Negative) by plotting the optimal hyperplane.

In addition to separating the instances, this method generates a hyperplane so that it's as far as possible from the case closest to each class. SVM and its modifications help solve complex tasks. The performance of the SVM classifier depends on the choice of the regularization kernel parameters [19], [20].

2) Decision tree classifier

Generally, the decision tree technique is used in binary classification problems. The objective is to build a set of choices in the form of a graphic tree composed of nodes and branches according to each collected attribute. The DT algorithm uses if-then rules to construct the classification model in the form of a tree structure [21]. The process continues by decomposing the data into smaller structures and possibly associating them with an incremental decision tree. Rules are learned sequentially using training data one at a time. Each time a rule is learned, the tuples covering the rules are deleted. The process continues on the training set until the endpoint is reached [22]. The final structure of the model looks like a tree with nodes and leaves. The probability *P(T)* of estimating that an observation *j* is in node *X* is given by equation (1):

$$P(T) = \sum_{j \in X} w_j \tag{1}$$

3) Random forest classifier

Random Forest (RF) is an efficient algorithm for prediction problems. Indeed, this is a particular case of bagging applied to decision trees. The principle of bagging methods is to calculate the average of the forecasts of several independent models to reduce the variance and the forecast error. To build a RF model, we select several bootstrap examples [23]. We use a large number of decision trees, each constructed with a different subsample of the training set. For each construction of a tree, the decision at a node is made according to a subset of variables drawn at random. Then, we use all the decision trees produced to make the prediction, with a majority vote for the classification, predicted variable of type factor [24]. Figure 2 illustrates the construction architecture of an RF model.

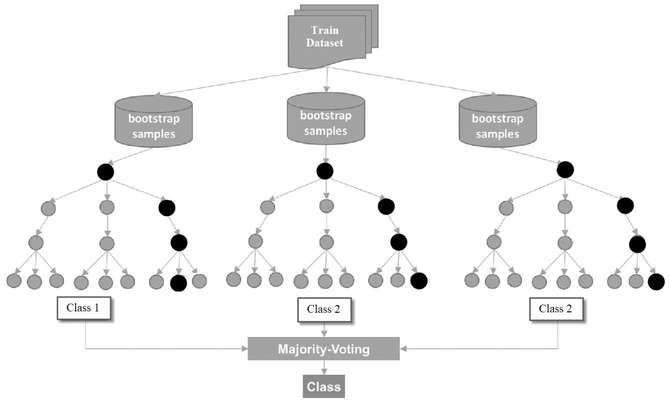

Fig. 2. Diagram representing the operating of the Random Forest algorithm.

4) *Adaboost classifier*

Adaptive Boosting (AdaBoost) is an algorithmic method known in the field of machine learning. This approach is the most widely used derivative of the Boosting method, which aims to stimulate the learning algorithm performance. AdaBoost consists of efficiently transforming a "weak" classifier into a "strong" classifier by reducing error rates. AdaBoost's algorithm calls a learning algorithm at each iteration that trains the instances to classify. Thereafter, he define a new probability distribution for the learning instances [25]. Based on the results of the algorithm in the previous iteration while increasing the weight of the instances that are not correctly classified. At the end, AdaBoost combines weak data with a weighted vote to deduce a strong classifier. In fact, AdaBoost is trying to find an optimal classifier from the combination of a weak learning data set [26]. The principle of Boosting is to reuse a classifier several times by assigning to the learning instances, each time, a different weighting and then combining the results in order to find a single "strong / very precise" classifier.

5) *XGBoost classifier*

The XGBoost algorithm is a real star in machine learning competitions. It is an optimized open source implementation of the gradient boost tree algorithm. Indeed, Gradient Boosting is a supervised learning algorithm. The principle is to combine the results of a simpler and weaker set of models in order to provide a better prediction. The idea is simple: instead of using a single model, the algorithm will use several, which will be combined to obtain a single result. The algorithm works sequentially. Unlike for example the random forest. This will make it slower but it will allow the algorithm to improve by capitalization compared to previous executions. He begins by building and evaluating a first model. From the evaluation of the first model, each individual will be weighted according to the performance of the prediction [27]. XGBoost contains a large number of modifiable and adjustable hyper-parameters to improve performance.

IV. Proposed Methodology

A. *Global overview*

In this paper, we aim to improve the performance of classification models by using the *GridSearch* function. Figure 3 illustrates the main stages to build the prediction system of patients infected with COVID-19.

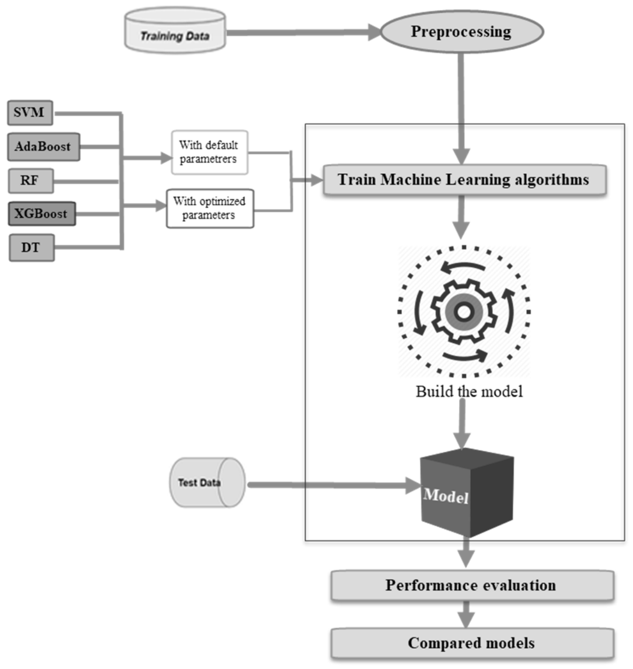

Fig. 3. Illustration of the proposed methodology steps.

The ratio used for the training and testing process is 80% and 20% respectively. We carried out two experiments. The first one, we trained and tested the models with their default hyper-parameters. Then, we calculated the performance metrics and drew the confusion matrix. In the second experiment, we examined the hyper-parameters and created grid parameters for each algorithm. This grid contains range of values necessary to find the optimal hyper-parameter value. With the same previous evaluation approach, we extracted the performances of the optimized models. Finally, we compared these performances with the results obtained in the first experiment.

B. *Preprocessing*

The main purpose of this step is to remove missing values and unnecessary features.

TABLE 1. The AEH Dataset Architecture

Instances	Positive : 1	558
	Negative : 0	702
Relevant Features	Input	17
	Output	2
Train set (80%)	Positive cases	436
	Negative cases	572
Test set (20%)	Positive cases	122
	Negative cases	130

We keep the significant and correlate features with the outcome. The correlation function make possible to generate a vector of important features [28]. Therefore, we performed a series of preprocessing operations, including encoding, removing unnecessary data and features engineering. Table 1 shows the final dataset architecture used

C. Parameter selection with GridSearch

Optimizing hyper-parameters is the process of adjusting a parameter to minimize the cost function of the model. In fact, this process refers to any parameter of a machine-learning model. Most models of machine learning require at least one variable defined before learning. For example, decision tree model, support vector machines [29], neural networks and regularization algorithms for ordinary least squares. The purpose of using *GridSearch* function is to define the optimal parameters of a model. This allows rapid iteration over possible combinations of values for any hyper-parameters value [30]. Indeed, the search in the grid works by trying all the possible parameters combinations. Table 2 represents the optimal hyper-parameters obtained by the grid search function. These configurations give the best results in terms of prediction accuracy.

TABLE 2. THE OPTIMAL HYPERPARAMETERS OBTAINED WHICH GIVE THE BEST RESULTS IN TERMS OF PREDICTION ACCURACY

Classifier	Hyper-Parameters	Value
SVM	Svc__C	1000
	Cache_size	20
	Gamma	0.0001
	Kernel	RBF
	Degree	3
AdaBoost	Algorithm	SAMME.R
	Learning_rate	1.0
	N_estimators	50
	Random_state	0
RF	Bootstrap	True
	Ccp_alpha	0.0
	Criterion	gini
	Max_depth	25
	Min_samples_leaf	1
	Min_samples_split	2
	N_estimators	100
XGBoost	Base_score	0.5
	Booster	Gbtree
	Learning_rate	0.1
	Subsample	0.5
	N_estimators	700
	Max_depth	6
DT	Min_samples_leaf	1
	Min_samples_split	2
	Splitter	Best
	Random_state	0

D. Performance Evaluation Methods

Performance evaluation model allows checking the accuracy and efficiency. There are many ways to assess a classifier. In this study, we used the *Holdout* method, which consists of dividing the dataset into two parts as a test, and train set 20% and 80% respectively. The train set is used to train the data and the invisible test set is used to test its predictive performance. Moreover, we used the Cross-validation method to avoid the problem of over-fitting [31]. Then, we calculated some evaluation metric namely the accuracy, precision, recall, ROC and the F1-score [32].

V. EXPERIMENTAL RESULTS AND DISCUSSION

In this section, we evaluate the performance of the five supervised machine learning algorithms using the *Holdout* and Cross-validation method. We assessed the model quality using different metrics under two experiments:

- Experiment 1: keep the default model hyper-parameters and use the AEL dataset with preprocessing.
- Experiment 2: Use the same dataset with the same preprocessing, but we develop for each model a *GridSearch* of the parameters to find the best combination.

A. Experiment 1: algorithms with default parameters

In this subsection, we present the performance results of the algorithms with default hyper-parameters. The five algorithms train on 1008 instances and then they test on 252. In order to draw the confusion matrix of each classification, we have calculated the elements of binary classification. Table 3 presents the values of cases where the prediction is positive and the real value is positive (True Positive). The cases where the prediction is positive but the real value is negative (False Positive). The cases where the prediction is negative and the real value is negative (True Negative). Moreover, the cases where the prediction is negative but the real value is positive (False Negative).

TABLE 3. CONFUSION MATRIX ELEMENTS REPRESENTING THE PERFORMANCE OF THE FIVE CLASSIFIER STUDIED WITH DEFAULT HYPER-PARAMETERS

Models	TP	FP	TN	FN
SVM	90	32	129	1
AdaBoost	91	31	122	8
RF	111	11	127	3
XGBoost	112	10	126	4
DT	109	12	117	13

From these results, we can see that the XGBoost, RF and DT model provided a good classification in comparison with the other models. To illustrate these performances well, we have calculated and presented in Table 4 some percentage performance metrics.

TABLE 4. PERFORMANCE EVALUATION METRIC OF THE FIVE ALGORITHMS WITH DEFAULT HYPER-PARAMETERS

Metrics	SVM	AdaBoost	RF	XGBoost	DT
Accuracy (%)	86.90	84.52	94.44	94.44	90.08
Precision (%)	98.90	91.92	97.37	96.55	89.43
Recall (%)	73.77	74.59	90.98	91.80	90.16
ROC	0.86	0.84	0.94	0.94	0.90
F1-score	0.84	0.82	0.94	0.94	0.89

Now, we can quickly notice that the three algorithms of the decision trees (XGBoost, RF, and DT) have an accuracy of 94%. The SVM and the AdaBoost does not exceed 86%. Yet the SVM model surpass all models in Precision, it reaches 98.90%. For the other metrics that remain recall, ROC and F1, we noticed that always the trees algorithm family dominates the high performance.

B. *Experiment 2: algorithms with optimized parameters*

In the second experiment, we trained the same algorithms on the same dataset, except that this time we introduced optimal hyper-parameters using the *GridSearch* function. We obtained different classifications from those in Experiment 1. Table 5 represents the classification elements of the five models.

TABLE 5. Confusion Matrix Elements Representing The Performance Of The Five Classifier Studied With Optimized Hyper-Parameters

Models	TP	FP	TN	FN
SVM	108	14	129	1
AdaBoost	113	9	122	8
RF	110	12	129	1
XGBoost	111	11	125	5
DT	109	13	120	10

From this table, we notice that the classification of the five models has been improved compared to the previous experiment. Table 6 includes the new values of the evaluation metrics of performance models.

TABLE 6. Performance Evaluation Metric Of The Five Algorithms With Default Hyper-Parameters

Models	SVM	AdaBoost	RF	XGBoost	DT
Accuracy (%)	94.05	93.25	94.84	94.44	90.87
Precision (%)	99.08	93.39	99.10	96.55	91.60
Recall (%)	88.52	92.62	90.16	91.80	89.34
ROC	0.93	0.93	0.94	0.95	0.90
F1-score	0.93	0.93	0.94	0.94	0.90

By comparing these performances with the results cited in Table 4, we observe that some models have experienced a slight improvement in performance. However, the SVM and AdaBoost models experienced a significant improvement in terms of the recall metric. Table 7 illustrates the percentage improvement of each model compared to experiment 1.

TABLE 7. Percentage Improvement Rate Of The Five Models Compared To Experiment 1

Metrics	SVM	AdaBoost	RF	XGBoost	DT
Accuracy (%)	**+7.15**	+8.73	+0.4	0	+0.79
Precision (%)	+0.18	+1.47	+1.73	0	+2.17
Recall (%)	**+14.7**	**+18.03**	-0.82	0	-0.82
ROC	+7.38	+9.01	+0.38	+0.8	+0.75
F1-score	+9	+1.35	+0.35	0	+0.66

The XGBoost model kept the same performances of experiment 1, except a slight improvement of 0.8% on the ROC. The RF and DT recorded a slight improvement in all metrics except the recall, which decreased with 0.82%.

On the other hand, the performance of both SVM and AdaBoost models has been greatly increased. AdaBoost's recall reached an extreme augmentation of 18.03% followed by SVM with 14.7% on the same metric.

C. *Discussion*

In this research, we conducted two experimental studies in order to improve the prediction of patients infected with COVID-19 using five machine-learning algorithms SVM, AdaBoost, RF, XGBoost and DT. We explored a public dataset containing information on blood tests of approximately 5,644 patients. We performed a series of pre-processing operations to extract the important features and avoid missing values. Finally, we obtained a dataset containing 1260 instances. In the two experiments, we trained the algorithms on the same separation ratio of the dataset (80% training and 20% for the test). In experiment 2, we use a *GridSearch* for each algorithm to find the optimal hyper-parameters. From the experimental results section, we observed that the performance of the majority of models has been improved. The AdaBoost model experienced a major increase in the recall metric. In addition, the SVM show a significant improvement on all metrics. A very important remark that we have drawn from the confusion matrices is that the two classification obtained by the SVM and RF models have a FN equal to one. Which means that these two models have an extreme sensitivity to identify the patients actually infected with COVID-19.

VI. Conclusion and Perspectives

In summary, the optimization of the hyper-parameter plays a major role in each algorithm. This can influence on the performance of the model learning. In this paper, we showed that the use of a grid search function improve and increase the performance of a model. This grid search finds the best configuration of an algorithm. This study could help future researchers to choose the optimal hyper-parameters to improve their model using one of the *GridSearch* functions.

In future work, we will develop an architecture of an optimized convolutional neural network model based on Transfer Learning algorithms. To deal with detection problems of patients infected with COVID-19 based on X-ray or CT-images.

Acknowledgment

This work is a part of a project supported by co-financing from the CNRST (Centre National pour la Recherche Scientifique et Technique) and the Hassan II University of Casablanca, Morocco. The project is selected in the context of a call for projects entitled “Scientific and Technological Research Support Program in Link with COVID-19” launched in April 2020 (Reference: Letter to the Director of “Ecole Normale Supérieure de l’Enseignement Technique de Mohammedia” dated June 10th 2020).

References

[1] J. Qiu, Q. Wu, G. Ding, Y. Xu, and S. Feng, “A survey of machine learning for big data processing,” *EURASIP J. Adv. Signal Process.*, vol. 2016, no. 1, p. 67, Dec. 2016, doi: 10.1186/s13634-016-0355-x.

[2] H. Moujahid, B. Cherradi, L. Bahatti, O. Terrada, and S. Hamida, “Convolutional Neural Network Based Classification of Patients with Pneumonia using X-ray Lung Images,” *Advances in Science, Technology and Engineering Systems Journal*, vol. 5, no.5, pp. 167–175, 2020, 10.25046/aj050522.

[3] O. Terrada, B. Cherradi, S. Hamida, A. Raihani, H. Moujahid, and O. Bouattane, “Prediction of Patients with Heart Disease using Artificial Neural Network and Adaptive Boosting techniques,” presented at the 2020 3rd International Conference on Advanced Communication

Technologies and Networking (CommNet), 2020, doi: 10.1109/CommNet49926.2020.9199620.

[4] S. Laghmati, B. Cherradi, A. Tmiri, O. Daanouni, and S. Hamida, "Classification of Patients with Breast Cancer using Neighbourhood Component Analysis and Supervised Machine Learning Techniques," presented at the 2020 3rd International Conference on Advanced Communication Technologies and Networking (CommNet), Marrakech, Morocco, Morocco, 2020, doi: 10.1109/CommNet49926.2020.9199633.

[5] N. Ait Ali, B. Cherradi, A. El Abbassi, O. Bouattane, and M. Youssfi, "GPU fuzzy c-means algorithm implementations: performance analysis on medical image segmentation," *Multimed Tools Appl*, vol. 77, no. 16, Art. no. 16, Aug. 2018, doi: 10.1007/s11042-017-5589-6.

[6] N. A. Ali, B. Cherradi, A. E. Abbassi, O. Bouattane, and M. Youssfi, "Modelling the behavior of the CPU and the GPU versus the clusters number variation for sequential and parallel implementations of BCFCM algorithm," *ARPN Journal of Engineering and Applied Sciences*, vol. 12, no. 21, Art. no. 21, 2017.

[7] N. A. Ali, B. Cherradi, A. El Abbassi, O. Bouattane, and M. Youssfi, "Parallel Implementation and Performance Evaluation of some Supervised Clustering Algorithms for MRI Images Segmentation," in *Proceedings of the 4th International Conference on Big Data and Internet of Things*, Rabat Morocco, Oct. 2019, pp. 1–7, doi: 10.1145/3372938.3373007.

[8] O. Bouattane, B. Cherradi, M. Youssfi, and M. O. Bensalah, "Parallel c-means algorithm for image segmentation on a reconfigurable mesh computer," *Parallel Computing*, vol. 37, no. 4, Art. no. 4, Apr. 2011, doi: 10.1016/j.parco.2011.03.001.

[9] H. Moujahid, B. Cherradi, and L. Bahatti, "Convolutional Neural Networks for Multimodal Brain MRI Images Segmentation: A Comparative Study," in *Smart Applications and Data Analysis*, vol. 1207, M. Hamlich, L. Bellatreche, A. Mondal, and C. Ordonez, Eds. Cham: Springer International Publishing, 2020, pp. 329–338, doi.org/10.1007/978-3-030-45183-7_25.

[10] O. Terrada, S. Hamida, B. Cherradi, A. Raihani, and O. Bouattane, "Supervised Machine Learning Based Medical Diagnosis Support System for Prediction of Patients with Heart Disease," *Advances in Science, Technology and Engineering Systems Journal*, vol. 5, no.5, pp. 269–277, 2020, 10.25046/aj050533.

[11] H. Song, I. Triguero, and E. Özcan, "A review on the self and dual interactions between machine learning and optimisation," *Prog Artif Intell*, vol. 8, no. 2, pp. 143–165, Jun. 2019, doi: 10.1007/s13748-019-00185-z.

[12] M. Feurer and F. Hutter, "Hyperparameter Optimization," in *Automated Machine Learning*, F. Hutter, L. Kotthoff, and J. Vanschoren, Eds. Cham: Springer International Publishing, 2019, pp. 3–33.

[13] I. Guyon ***et al.***, "Analysis of the AutoML Challenge Series 2015–2018," in *Automated Machine Learning*, F. Hutter, L. Kotthoff, and J. Vanschoren, Eds. Cham: Springer International Publishing, 2019, pp. 177–219.

[14] James Bergstra, "Random Search for Hyper-Parameter Optimization," *Journal of Machine Learning Research*, 2012.

[15] H. A. Fayed and A. F. Atiya, "Speed up grid-search for parameter selection of support vector machines," *Applied Soft Computing*, vol. 80, pp. 202–210, Jul. 2019, doi: 10.1016/j.asoc.2019.03.037.

[16] S. Kaur, H. Aggarwal, and R. Rani, "Hyper-parameter optimization of deep learning model for prediction of Parkinson's disease," *Machine Vision and Applications*, vol. 31, no. 5, p. 32, Jul. 2020, doi: 10.1007/s00138-020-01078-1.

[17] A. H. Victoria and G. Maragatham, "Automatic tuning of hyperparameters using Bayesian optimization," *Evolving Systems*, May 2020, doi: 10.1007/s12530-020-09345-2.

[18] P. Lameski, E. Zdravevski, R. Mingov, and A. Kulakov, "SVM Parameter Tuning with Grid Search and Its Impact on Reduction of Model Over-fitting," in *Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing*, vol. 9437, Y. Yao, Q. Hu, H. Yu, and J. W. Grzymala-Busse, Eds. Cham: Springer International Publishing, 2015, pp. 464–474.

[19] I. Syarif, A. Prugel-Bennett, and G. Wills, "SVM Parameter Optimization using Grid Search and Genetic Algorithm to Improve Classification Performance," *TELKOMNIKA*, vol. 14, no. 4, p. 1502, Dec. 2016, doi: 10.12928/telkomnika.v14i4.3956.

[20] R. Liu, E. Liu, J. Yang, M. Li, and F. Wang, "Optimizing the Hyper-parameters for SVM by Combining Evolution Strategies with a Grid Search," in *Intelligent Control and Automation*, vol. 344, D.-S. Huang, K. Li, and G. W. Irwin, Eds. Springer Berlin Heidelberg, 2006, pp. 712–721.

[21] S. B. Kotsiantis, "Decision trees: a recent overview," *Artif Intell Rev*, vol. 39, no. 4, pp. 261–283, Apr. 2013, doi: 10.1007/s10462-011-9272-4.

[22] T. Shaikhina, D. Lowe, S. Daga, D. Briggs, R. Higgins, and N. Khovanova, "Decision tree and random forest models for outcome prediction in antibody incompatible kidney transplantation," *Biomedical Signal Processing and Control*, vol. 52, pp. 456–462, Jul. 2019, doi: 10.1016/j.bspc.2017.01.012.

[23] M. M. Ramadhan, I. S. Sitanggang, F. R. Nasution, and A. Ghifari, "Parameter Tuning in Random Forest Based on Grid Search Method for Gender Classification Based on Voice Frequency," *dtcse*, no. cece, Oct. 2017, doi: 10.12783/dtcse/cece2017/14611.

[24] S. Wang, W. Ren, Y. Zhang, and F. Liang, "Random Forest Classifier for Distributed Multi-plant Order Allocation," in *Proceeding of the 24th International Conference on Industrial Engineering and Engineering Management 2018*, G. Q. Huang, C.-F. Chien, and R. Dou, Eds. Singapore: Springer Singapore, 2019, pp. 123–132.

[25] R. Wang, "AdaBoost for Feature Selection, Classification and Its Relation with SVM, A Review," *Physics Procedia*, vol. 25, pp. 800–807, 2012, doi: 10.1016/j.phpro.2012.03.160.

[26] J. Wang, L. Gao, H. Zhang, and J. Xu, "Adaboost with SVM-Based Classifier for the Classification of Brain Motor Imagery Tasks," in *Universal Access in Human-Computer Interaction. Users Diversity*, vol. 6766, C. Stephanidis, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 629–634.

[27] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco California USA, Aug. 2016, pp. 785–794, doi: 10.1145/2939672.2939785.

[28] B. Xue, M. Zhang, W. N. Browne, and X. Yao, "A Survey on Evolutionary Computation Approaches to Feature Selection," *IEEE Trans. Evol. Computat.*, vol. 20, no. 4, pp. 606–626, Aug. 2016, doi: 10.1109/TEVC.2015.2504420.

[29] Y. Bao and Z. Liu, "A Fast Grid Search Method in Support Vector Regression Forecasting Time Series," in *Intelligent Data Engineering and Automated Learning – IDEAL 2006*, vol. 4224, E. Corchado, H. Yin, V. Botti, and C. Fyfe, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 504–511.

[30] Á. B. Jiménez, J. L. Lázaro, and J. R. Dorronsoro, "Finding Optimal Model Parameters by Discrete Grid Search," in *Innovations in Hybrid Intelligent Systems*, vol. 44, E. Corchado, J. M. Corchado, and A. Abraham, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 120–127.

[31] H. Wang and H. Zheng, "Model Validation, Machine Learning," in *Encyclopedia of Systems Biology*, W. Dubitzky, O. Wolkenhauer, K.-H. Cho, and H. Yokota, Eds. New York, NY: Springer New York, 2013, pp. 1406–1407.

[32] H. Dalianis, "Evaluation Metrics and Evaluation," in *Clinical Text Mining*, Cham: Springer International Publishing, 2018, pp. 45–53.