Parallel, Distributed and Federated Learning for applying PRIM in Big Data

Rym Nassih
Phd Student in Machine Learning
Research team AMIPS
Ecole Mohammadia d'Ingénieurs
Mohamed V University in
Rabat, Morocco
rymnassih@research.emi.ac.ma

Abdelaziz Berrado
Professor of Industrial Engineering
Research team AMIPS
Ecole Mohammadia d'Ingénieurs
Mohamed V University in
Rabat, Morocco
berrado@emi.ac.ma

Abstract

Training models in machine learning often requires big computational capacities, even more when it involves Big Data. The Patient Rule Induction Method (PRIM), introduced as a bump hunting algorithm and a subgroup discovery method, already consumes a lot of computing resources when searching for the rules. In this computational research area, several methods have been developed to handle the problem of computing a big dataset: parallel learning, distributed learning and federated learning. In this paper, we investigate the literature review of these three methods to define each one and the differences between them, and then we determine which of these methods is best suited for the implementation of PRIM in big dataset for out futur works in Big Data.

Keywords
Parallel learning, Distributed Learning, Federated Learning, Big Data, Patient Rule Induction Method.

1. Introduction

Everyday a huge amount of data is generated forming at a large-scale Big Data (Sagiroglu and Sinanc, 2013). These data are characterized by the commonly known five Vs: Volume, Variety, Value, Veracity and Velocity. The Volume of big Data mainly brings up the issue of imbalanced data between the classes in a binary classification problem. The Variety of big Data means that a data has multiple sources. The Value, often considered as the most important V, refers to the explainability of the knowledge induced from mining the data for the domain's problem it's used for. The Veracity refers to the truthfulness of the data in the dataset, or as considered in today's machine learning literature, the fairness of the data collection and, thus, the fairness of the model trained on this data. And finally, the Velocity indicates the frequency of the data collection and its influence on the Volume, hence on the model and the other Vs.

In order to use machine learning algorithms to build models on big data, the main problem is the volume. From supervised to unsupervised learning, deep learning or reinforcement learning, using any of these machine learning approaches can lead to hours of processing and calculations, and make testing, evaluating and improving a model nearly impossible. To palliate to this issue, researches have investigates throughout the decades, several ways to generate the results. Among those techniques the most popular and used ones are the distributed learning, the parallel

learning and later, the introduction of federated learning. Their aim is to speed up the process without losing scalability and performances, and later with federated learning, the main focus was made on the data confidentiality. Moreover, it facilitates obtaining a global model without centralizing privacy-sensitive data, thereby contributing to the development of trustworthy AI systems. This form of parallel, distributed, and federated machine learning has gained substantial interest in recent years, both from researchers and practitioners, and may allow for disruptive changes in areas such as smart assistants, machine learning on medical or industrial data, and autonomous driving.

In the literature, several articles raise the problem of the big data implementation (Skënduli et al.(2018)). (Ducange et al, 2020) overview the recent research in generating fuzzy rules classification in Big Data using distributed learning by comparing seven algorithms on a MapReduce framework for distributed learning. Verbraeken et al, (2020), give a survey on distributed machine learning and some applications. Based on the MapReduce paradigm, Dean and Ghemawat (2008) have proposed new distributed implementations of data mining and machine learning for Big Data. Likewise, Ludwig (2015) and Kim et al(2014) both investigates the design, the implementation and experiment clustering algorithms in a distributed machine learning context. As for classification algorithms, Bechini et al (2016) and Maillo et al(2017) discuses very interesting results, in terms of accuracy and scalability. In his work, Zhou et al(2017) highlights current advancements, problems, and goals in designing, implementing, and deploying data mining and machine learning algorithms for Big Data.

As stated before, the classical data storage and elaboration paradigms are not suitable for handling Big Data. Specially when the algorithm involved is already consuming time and computational resources in classical datasets. Indeed, the Patient Rule Induction Method, introduced later in the paper, is already a method that mines rules in the designed subset slowly and «patiently ». The fact that it only peeling 5% of the data makes the procedure slow and heavy to compute. Although this method is originally adapted for high dimensional data, working with Big Data only increases the complexity.

1.1. Objectives

The objectives of our paper are, first, to investigate the distributed learning, the parallel learning and the federated learning in the literature in order to determine in which case everyone is used. Second, we aim at listing the different implementation of PRIM and the computational problems related to it, that could be solved by using one of the three techniques. And finally, we compare each approach in the case of building a classifier with PRIM to direct us for our ongoing and future works, since we will use Big Data in our study.

2. Overview of Parallel, Distributed and Federated Learning

With the limitations of the sequential processing, researchers have looked in other ways to compute machine learning algorithms and their hyperparameters in big data, thereby using parallel computing Padua D (2011). Philip K. Chan, and Salvatore J., Stolfo.(1993) introduced the concept of meta-learning, which is learning from models that have already learned from dataset, to reduce the computational time and capacity. Thereby, the concept of parallel learning took more sense with more implementation, by first splitting the dataset into several partitions and after, parallelizing the algorithm to find the best hyperparameters combinations.

In computer science, a system is said to be parallel when multiple processors have direct access to the same shared memory, and thereby they have the same address space, as illustrated in Figure 1. The multi-processors are all combined to solve the same problem, that is split in different tasks and executed simultaneously in parallel. Thanks to this method, a significant boost in performance can be achieved. It's opposed to traditional computing where we use only one processor for all the calculations, which raise the problem of the processor's limitations: speed, heat and performance. The parallel system is designed to speed up the execution of the program into multiple fragments and processing the fragments simultaneously. It is simultaneous use of multiple resources to solve the computational problem: to be run using multiple CPUs, a problem is broken into discrete parts that can be solved concurrently, and each part is further broken down to series of instructions, instructions from each part executes simultaneously on different CPUs.

To compute such a parallel procedure, parallelizing the algorithm is the most crucial. One should determine, according to the algorithm, which steps should be parallelized. In the case of splitting the dataset into different partitions, there are different packages and frameworks. On python, the module "multiprocessing" allows the user to fully leverage multiple processors on a single machine. We also find MPI (Gabriel et al. (2004)), OpenSHMEM (Chapman et al. (2010)) and Charm++(Kale and Krishnan (1993)), three different frameworks studied in Gu and Becchi (2019) to compare their performance, scalability, and load-balancing capability. On the other hand, Li et al.

(2017), propose an approach to build a parallel system allowing learning system improved by self-boosting based on the concepts of descriptive learning, predictive learning, and prescriptive learning into a uniform framework. They define a new paradigm where data, knowledge and action are taken as a whole system.

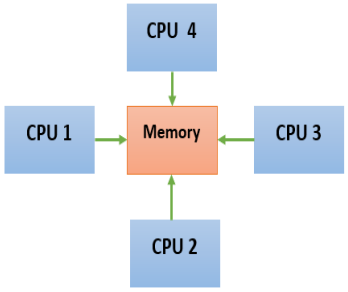

Fig 1: Parallel computing in the case of one memory for the parallel learning procedure

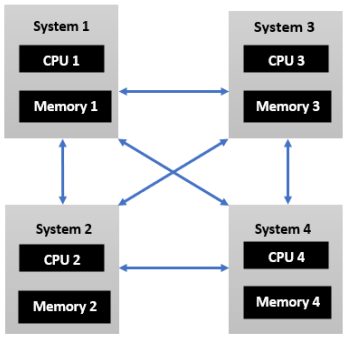

Fig 2: Distributed computing for the distributed learning procedure

Distributed machine learning refers to multimode machine learning algorithms and systems that are designed to improve performance, increase accuracy, and scale to larger input data sizes. Increasing the input data size for many algorithms can significantly reduce the learning error and can often be more effective than using more complex methods (Peteiro-Barral et al. (2011)). Distributed machine learning allows companies, researchers, and individuals to make informed decisions and draw meaningful conclusions from large amounts of data.

Distributed machine learning is applied on a distributed system, which is a multi-nodal system that builds training models by independent training on different nodes, as illustrated in Figure 2. Having a distributed training system accelerates training on huge amounts of data. When working with big data, training time exponentially increases which makes scalability and online re-training. For example, let's say we want to build a recommendation model, and based on the user interaction everyday, we wish to re-train the models. We could see the user-interaction as high as hundreds of clicks per user and millions of users. In this case there is an enormous amount of data that needs to be used to train the recommendation model on a daily basis. This is where distributed learning systems can be built and training can be parallelized to optimize time. Table 1 gives an overview on the comparison between parallel system and distributed system in computing.

Table 1. Comparison between parallel and distributed system on some criteria for computing

	PARALLEL SYSTEM	**DISTRIBUTED SYSTEM**
MEMORY	Tightly coupled system, shared memory	Loosely coupled system, distributed memory
PROCESSOR INTERCONNECTION	Order of Terabytes	Order of Gigabytes
MAINFOCUS	-Performance: cost and scalability -Scientific Computing	-Performance: cost and scalability -Reliability -Resource sharing

Federated learning was defined first by McMahan et al. (2016) and then presented more clearly in a Google's workshop. Compared to distributed learning, federated learning algorithms are fundamentally different and are primarily for addressing data privacy. In a traditional data science pipeline, the data is collected to a single server and used to build and train a centralized model. In effect, federated learning is having a centralized model using decentralized model training. In federated learning systems, a seed parameter set is sent to independent nodes containing data and the models are trained on the local nodes using data stored in these respective nodes. Once the model is trained independently, each of these updated model weights are sent back to the central server where they are

combined to create a highly efficient model. Using global data training improves the model efficiency by a big factor. This also ensures that the data in each node adheres to data privacy policies and protects any data leak/breach. As an example, we can look at an autocomplete model. The user data on phone/tablet/laptop can be confidential and the users would not want their data to be centrally stored anywhere. In this case, FL can be used to build a central autocomplete model which is effectively being trained (by combining independent trained model weights) on millions of users on their devices. Figure 3 illustrates how the federated learning works. Wrapping up, we can say that distributed learning is about having centralized data but distributing the model training to different nodes, while federated learning is about having decentralized data and training and in effect having a central model. In the literature, several works have been done using federated learning, Li et al. (2020), give a review of some applications in federated learning. According to Yang et al.(2019) FL is divided into three categories: horizontal FL, vertical FL and federated transfer learning. The horizontal FL concerns the data that differs from device to devices but still has the same features. The vertical FL concerns the data that overlaps on different devices, it has the same ID, but different features, a union of them compose one example of the overall dataset. But in real life scenarios, data neither share the same sample space or feature. In this case, transfer learning enables to move the knowledge from a source domain to a target domain.

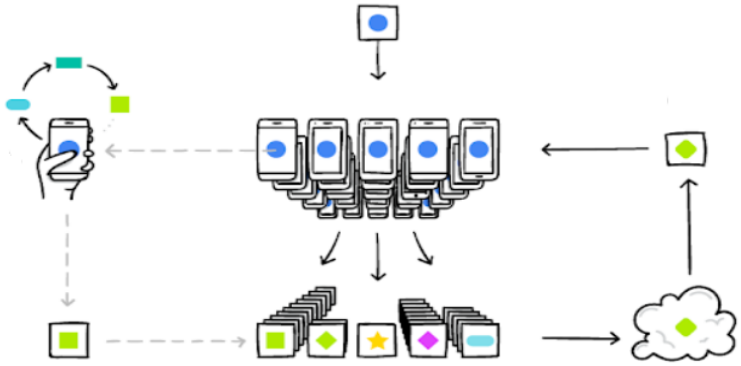

Figure 3: Illustration of Federated learning in a simple manner. The device here, the phone, based on the usage data, runs a model that is added to others from other devices, and the aggregation of these models is then implemented in the devices, to restart the loop and be in a continuous amelioration of the models and the system of recommendations.

3. Patient Rule Induction method

3.1. Overview

The Patient Rule Induction Method (PRIM) is a Bump Hunting algorithm introduced by Friedman and Fisher, to innovate in the search strategy by proceeding with bumps [32]. PRIM generates a list of rules that are rectangular in the space, hence the box induction. It consists of two phases. The first step, called the top-down peeling consists of the box construction. Given a subset of the input variables subspace, each box is delimited by the variables defining the chosen subspace. At each peel the portion of data removed can be controlled by the user and is usually 5%, hence the term patient. The final box found after the peeling process may not be optimal because of past greedy suboptimal choices. Therefore, the second step of PRIM is the bottom up pasting in which the boxes are expanded by iteratively enlarging their boundaries as long as the outcome's density increases. Figure 4 illustrates PRIM's process to find a box in the first phase of the top-down peeling. These two steps are repeated recursively to find other regions until stopping criteria are reached. Thus, PRIM is a greedy algorithm.

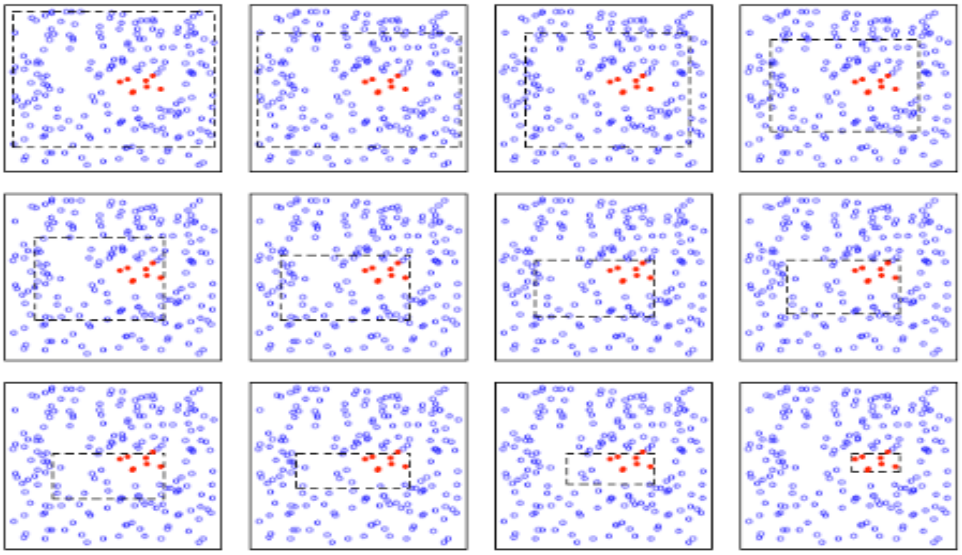

Fig 4: This figure shows the procedure of the first phase of PRIM, the top-down peeling, to find one box. Indeed, the algorithm peels the space dimension by dimension with checking the thresholds given and the satisfaction of the target variable until reaching the bump and where the size is greater than the threshold.

The outcome of PRIM is a set of boxes/regions that define rules relating the targeted label of the depended variable to the explanatory attributes in the data subspace chosen initially by the analyst as shown in Figure 5.

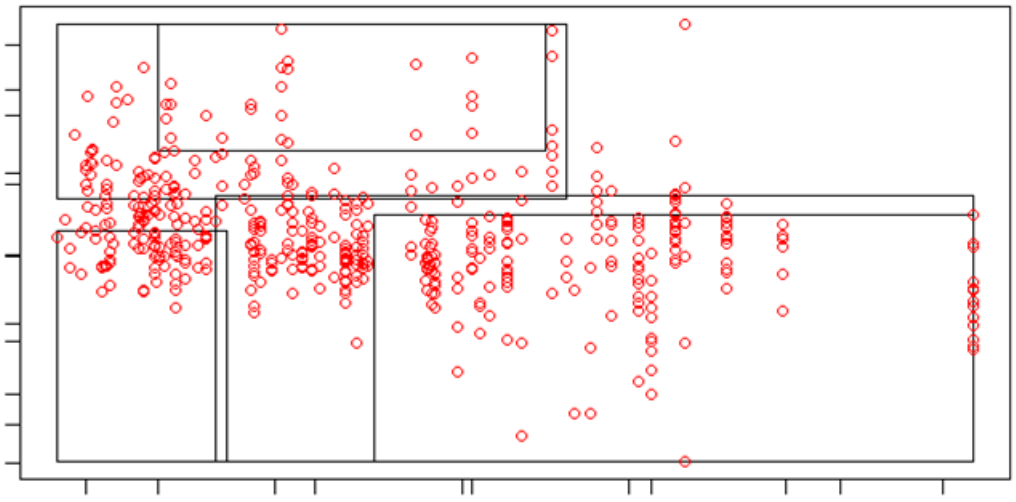

Figure 5: Illustration of a box induction in the search space deigned by the user

3.2. Related Work

Nassih and Berrado (2019) give a literature review PRIM in the research area. Polonik & Wang (2010) propose a modification of the second phase of the algorithm, and Chong, I. G., & Jun, C. H. (2008) who introduce a PRIM-like method specially to deal with ordinal discrete variables. Then in Díaz-Pachón et al.(2014) the PRIM is mixed with the Principle Component Analysis to reduce the search space and hence give a much more optimal result. Besides these modifications and some listed in Nassih and Berrado (2019), the majority of the application of PRIM show the same inconvenient: PRIM consumes a lot of computing resources and time. Indeed, in Nannings et al. (2008) Sadiq et al. (2017), Kaveh et al. (2018) and Dyson et al. (2007), beside the algorithm

performance that scores very well in comparison with some other popular methods as CART, the authors all stressed out the time and the computing resources needed to achieve a satisfactory outcome. For these reasons, we aim to dress a guideline for future works in big data.

4. Case of PRIM for big data applications

Performing machine learning algorithms in a distributed environment first involves conceptually converting single-threaded algorithms to parallel algorithms. Diverse parallel techniques are used for the distributed computing process. This step can often be the most difficult because it is algorithm-specific and requires that the user has a strong understanding of the underlying algorithm. Whether we are going to use distributed systems or parallel systems to compute PRIM on a large-scale dataset, parallelizing the algorithm is the very first step. PRIM has got some meta-parameters to tunes: the peeling criteria, the pasting criteria and the minimum support. To find the optimal combination, experts usually compute a lot of possibilities different chosen subsets.

There are different ways to parallelize the algorithm. The first one concerns the parallelization of the database. Indeed, by setting a combination of meta parameters, one can partition the dataset into several on the present nodes, and thus run the algorithm on each partition independently of the others. In order to recover the most optimal result, one will have to group all the boxes found and use either an optimization method such as metarules (Berrado and Runger (2007)) to create rule association between the rules and thus reducing the set of the rules, or use the principle of Ensemble Learning (Dong et al.(2020)), and create for every partition a model and then do the average of them all to obtain an optimal model for the whole dataset.

The second way of parallelizing PRIM, concerns the parallelization of the different combinations of meta-parameters as well as the different dataset partitions. In the case where the expert has a large computing capacity, especially in the case of a distributed computing system, the expert can select a few possible combinations of meta-parameters that he considers the most suitable to obtain the best rules, and thus create a list of tests of combinations on each partition. Although the computation time will be high, at the end of the process, the expert will have not only the exhaustive set of rules found, but also the performances for each combination of meta parameters.

The last way to parallelize PRIM would be to parallelize the different search subspaces. Indeed, one of the particularities of this algorithm is that the expert chooses upstream the attributes that will represent the search space of the rules. Moreover, in the case where the expert would like to test several search possibilities, we could apply the same approach as for the search of the optimal meta parameters. Note that in the case where one would like to test several combinations of subspaces as well as several combinations of meta parameters, the complexity of the learning process increases and would risk losing its scalability. In this case, we recommend a feature engineering, feature selection and even an exploratory step of the dataset as well as an in-depth analysis of the domain problematic in order to identify the few possible combinations or subspaces and to reduce the complexity.

5. Conclusion

In this paper, we give an overview of the three used techniques for computing a large-scale dataset in machine learning: parallel learning, distributed learning and federated learning. Each one of these techniques, is best suited in a case and addresses a different issue from the others. In the case of the Patient Rule Induction Method, we have three main components to use it for machine learning: choosing the search space, choosing the meta parameters and working with a high dimensional big dataset. In this case, and according to our review, we highlight that the application of PRIM on big data will depend on the domain. Indeed, if the experts already know which subset and meta parameters are the best for the rule search, the parallelization will concern only the data which will be split into different partitions. And then, to recover the overall model, one could consider it as an ensemble learning, or can consider all the boxes and then turn to pruning techniques. In the case of searching for the optimal meta-parameters' combination, the parallelization will concern computing in each node a partition on a combination. Therefore, the complexity will increase because on each node there will be a partition and a list of meta-parameters' combination possible. And the third case, concerns the search subspace. Again, the complexity increases and the time computing too, which lead us to stress the importance of the domain analysis for any size of data in machine learning. As for the Federated Learning, the use of it depends on the data policies concerning the privacy.

References

Bechini, A., Marcelloni, F. and Segatori, A. A., MapReduce solution for associative classification of big data. *Inf Sci. 2016*;332:33–55.

Berrado, A., & Runger, G. C. (2007). Using metarules to organize and group discovered association rules. *Data mining and knowledge discovery*, *14*(3), 409-431.

Chong, I. G., & Jun, C. H. (2008). Flexible patient rule induction method for optimizing process variables in discrete type. *Expert Systems with Applications*, *34*(4), 3014-3020.

Dean J., Ghemawat S., MapReduce: simplified data processing on large clusters. *Comm ACM*. 2008;51(1):107–13.

Díaz-Pachón, D. A., Dazard, J. E., & Rao, J. S. (2017). Unsupervised bump hunting using principal components. In *Big and Complex Data Analysis* (pp. 325-345). Springer, Cham.

Dong, X., Yu, Z., Cao, W. *et al.* A survey on ensemble learning. *Front. Comput. Sci.* 14, 241–258 (2020). https://doi.org/10.1007/s11704-019-8208-z

Ducange, P., Fazzolari, M. and Marcelloni, F., An overview of recent distributed algorithms for learning fuzzy models in Big Data classification. *J Big Data 7*, 19 (2020). https://doi.org/10.1186/s40537-020-00298-6

Dyson, G., Frikke-Schmidt, R., Nordestgaard, B. G., Tybjærg-Hansen, A., & Sing, C. F. (2007). An application of the patient rule-induction method for evaluating the contribution of the Apolipoprotein E and Lipoprotein Lipase genes to predicting ischemic heart disease. *Genetic Epidemiology: The Official Publication of the International Genetic Epidemiology Society*, *31*(6), 515-527.

Friedman, J. H., & Fisher, N. I. (1999). Bump hunting in high-dimensional data. *Statistics and computing*, *9*(2), 123-143.

Gabriel, E., Fagg, G. E., Bosilca, G., Angskun, T., Dongarra, J. J., Squyres, J. M., ... & Woodall, T. S. (2004, September). Open MPI: Goals, concept, and design of a next generation MPI implementation. In *European Parallel Virtual Machine/Message Passing Interface Users' Group Meeting* (pp. 97-104). Springer, Berlin, Heidelberg.

Gu, R., & Becchi, M. (2019, April). A comparative study of parallel programming frameworks for distributed GPU applications. In *Proceedings of the 16th ACM International Conference on Computing Frontiers* (pp. 268-273).

Kale, L. V., and S. Krishnan, "CHARM++: a portable concurrent object oriented system based on C++," *presented at the ACM Sigplan Notices*, 1993.

Kaveh, A., Hamze-Ziabari, S. M., & Bakhshpoori, T. (2018). Patient rule-induction method for liquefaction potential assessment based on CPT data. *Bulletin of Engineering Geology and the Environment*, *77*(2), 849-865.

Kim, S. H., Lee, D. H., & Kim, K. J. (2022). EWMA-PRIM: Process optimization based on time-series process operational data using the exponentially weighted moving average and patient rule induction method. *Expert Systems with Applications*, *195*, 116606

Kim,Y., Shim, K., Kim, M-S., and Lee, JS., DBCURE-MR: an efficient density-based clustering algorithm for large data using mapreduce. *Inf Syst. 2014*;42:15–35.

Lazarevic, A., & Obradovic, Z. (2002). Boosting algorithms for parallel and distributed learning. *Distributed and parallel databases*, *11*(2), 203-229.

Lee, D. H., Yang, J. K., & Kim, K. J. (2018). Dual-response optimization using a patient rule induction method. *Quality Engineering*, *30*(4), 610-620.

Lee, M. S., & Kim, K. J. (2008). MR-PRIM: patient rule induction method for multiresponse optimization. *Quality Engineering*, *20*(2), 232-242

Ludwig SA., Mapreduce-based fuzzy c-means clustering algorithm: implementation and scalability. *Int J Mach Learn Cybern*. *2015*; 6(6):923–34.

Maillo,J., Ramírez, S., Triguero, I., Herrera, F., KNN-IS: an iterative spark-based design of the k-nearest neighbors classifier for big data. *Knowl Based Syst. 2017*;117:3–15.

McMahan,H.B., Moore,E., Ramage,D., Hampson, S., and Aguera y Arcas,B., Communication-efficient learning of deep networks from decentralized data. *In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, pages 1273–1282, 2017 (original version on arxiv Feb. 2016).

Nannings, B., Abu-Hanna, A., & de Jonge, E. (2008). Applying PRIM (Patient Rule Induction Method) and logistic regression for selecting high-risk subgroups in very elderly ICU patients. *International Journal of Medical Informatics*, *77*(4), 272-279.

NASSIH, R., & BERRADO, A. Towards a patient rule induction method-based classifier. In *2019 1st International Conference on Smart Systems and Data Science (ICSSD)* (2019, October). (pp. 1-5). IEEE.

Padua D (2011) Encyclopedia of parallel computing. *Springer,* Berlin

Peteiro-Barral, D., Guijarro-Berdiñas, B., & Pérez-Sánchez, B. (2011, January). DEALING WITH "VERY LARGE" DATASETS-An Overview of a Promising Research Line: Distributed Learning. In *International Conference on Agents and Artificial Intelligence* (Vol. 2, pp. 476-481). SCITEPRESS.

Philip K. Chan, and Salvatore J., Stolfo. 1993. Toward parallel and distributed learning by meta-learning. *In Proceedings of the 2nd International Conference on Knowledge Discovery in Databases (AAAIWS'93).* AAAI Press, 227–240.

Polonik, W., & Wang, Z. (2010). PRIM analysis. *Journal of Multivariate Analysis*, *101*(3), 525-540.

Sadiq, S., Tao, Y., Yan, Y., & Shyu, M. L. (2017, April). Mining anomalies in medicare big data using patient rule induction method. In *2017 IEEE third international conference on multimedia Big Data (BigMM)* (pp. 185-192). IEEE.

Sagiroglu,S., and Sinanc,S., Big data: A review, *2013 International Conference on Collaboration Technologies and Systems (CTS)*, 2013, pp. 42-47, doi: 10.1109/CTS.2013.6567202.

Skënduli, M. P., Biba, M., & Ceci, M. (2018). Implementing Scalable Machine Learning Algorithms for Mining Big Data: A State-of-the-Art Survey. In *Big Data in Engineering Applications* (pp. 65-81). Springer, Singapore.

Verbraeken, J., Wolting, M., Katzy, J., Kloppenburg, J., Verbelen, T., and Rellermeyer, J. S. (2020). A survey on distributed machine learning. *ACM Computing Surveys (CSUR)*, *53*(2), 1-33.

Zhou,L., Pan,S., Wang, J., Vasilakos, AV., Machine learning on big data: opportunities and challenges. *Neurocomputing.* 2017;237:350–61.

Biographies

Rym Nassih is a fifth year Phd student in the field of Data Mining and Machine Learning in EMI School of Engineering at Mohammed V University in Rabat in Morocco. Her doctoral main research investigates data mining algorithms and machine learning models. She holds an engineering degree in Business Intelligence and Data Science from the INSEA, National Institute of Statistics and Applied Economy in Rabat in Morocco.

Dr. Abdelaziz Berrado is a full Professor of Industrial Engineering at EMI School of Engineering at Mohammed V University in Rabat in Morocco. He holds a Ph.D. degree in Decision Systems and Industrial Engineering from the Ira A. Fulton School of Engineering at Arizona State University. His research, teaching and consulting interests are in the areas of Big Data Analytics, Industrial Statistics, Operations and Supply Chain Modelling, Planning and Control with applications in healthcare, education and other industries. He focuses on developing frameworks, methods and tools for systems" diagnostics, optimization and control with the aim of operational excellence. He published several papers in research journals and conferences with local and international funding. He reviews for many journals and is member of INFORMS, IEOM and IEEE. In addition to academic work, he interacts continuously with different Industries in the areas of Machine Learning, Quality Engineering and Supply Chain Management. He was also a senior engineer at Intel.