Application of Machine Learning Algorithms for Predicting Energy Consumption of Servers

Meryeme EL YADARI[1*], Saloua EL MOTAKI[2], Ali YAHYAOUY[3],
Khalid EL FAZAZY[4], Hamid GUALOUS[5], Stéphane LE MASSON[6]
Computer Science, Sidi Mohamed Ben Abdellah University, Fes, Morocco[1, 3, 4]
Electrical Engineering, Caen Normandy University, Saint-lô, France[1, 5]
Computer Science, Chouaib Doukkali University, El Jadida, Morocco[2]
LaMSN - The House of Digital Sciences, USPN, Paris, France[3]
Orange Lab's, Lannion, France[6]

Abstract—**Energy management in data centers is currently a major challenge and arouses considerable interest. Many data center operators are seeking solutions to reduce energy consumption. In this work, the problem of resource overutilization-defined as the excessive usage of critical server resources such as CPU, RAM and storage surpassing their optimal capacity-in data centers is addressed, with a particular focus on servers. Estimating the energy consumption of servers in data centers allows its managers to allocate the necessary resources to ensure adequate quality of service. The research involved generating workloads performance on various servers, each connected to a wattmeter for energy consumption measurement. Data on resource utilization rates and server energy consumption were stored and analyzed. Machine learning models were then used to forecast server energy consumption. Parametric, non-parametric, and ensemble methods were employed and validated using accuracy measurements, non-parametric tests, and model complexity to assess the quality of energy consumption prediction models. The results demonstrated that certain models could provide predictions with a low margin of error and minimal complexity like polynomial regression, while other models showed lower performance. A comparative analysis is conducted to evaluate the performance and limitations of each approach.**

Keywords*—*Data center; server; machine learning; energy consumption; parametric methods; ensemble methods

I. Introduction

Data centers are centralized collections where networked computers provide computational resources for various applications such as web hosting, e-commerce, grid computing, cloud computing, and social networks [1]. The computing equipment also enables the processing, storage, and analysis of data within the data center. A data center is used in the form of a cloud computing infrastructure that offers enough storage capacity and computing for Internet of Things (IoT) services and managing real time massive IoT data generated by numerous IoT devices. However, to ensure optimal performance and high reliability, it is common for the data center network to be oversized, resulting in relatively low utilization of links in real life scenarios based on empirical observations [2]. Cooling or power interruptions can have significant repercussions on the environment of a data center. Since system failures can be extremely costly, it is crucial to implement reliability and redundancy measures. Redundancy in cooling systems helps minimize the risk of failure and improves performance by reducing downtime for maintenance and repairs. The primary advantage of redundancy is to increase system reliability [3]. Energy management is an important process that requires a systemic approach by addressing both the energy consumption of idle resources and the supporting infrastructure. However, truly sustainable operation should not only focus on energy management but also include investments in research and development of green energies, the utilization of renewable energy sources, and active measures to preserve the environment [4].

The data center infrastructure can be divided into three areas: the computer room, the support zone, and accessory spaces. The computer room is a space with controlled environmental conditions designed to accommodate equipment as well as cables directly linked to computer and telecommunications systems that generate a significant amount of heat. Additionally, Information Technology (IT) equipment is extremely susceptible to changes in humidity and temperature, so a data center must maintain constrained conditions to ensure the reliability and proper functioning of the equipment it contains. The support zones are where a variety of systems such as Uninterruptible Power Supplies (UPS), cooling control systems, and communication panels are situated. Finally, the accessory sections primarily consist of offices, a lobby, and restrooms [5].

The massive energy consumed by servers in data centers poses a major problem that requires strategic research to address such as high energy costs, environmental impact, and resource overutilization. Estimating server energy consumption is essential for efficient energy management, providing the necessary power to IT infrastructure, managing data center resources, and minimizing costs. The goal is to estimate servers' energy consumption using ML algorithms and build models with a low margin of error and weak complexity, which refers to the time required for model creation. Three methods, namely parametric, non-parametric, and ensemble methods, are employed for this purpose. Parametric methods rely on assumptions regarding the data's underlying distribution and use a fixed number of parameters to describe it. Non-parametric techniques do not presume any specific distribution about the distribution of data. They are flexible and can handle complex relationships without specifying a fixed number of parameters. Ensemble methods integrate several models to enhance predictive quality. They aggregate the predictions of several base models to produce a more accurate and robust prediction. Different models from each method were employed to assess the performance of each model and conduct a comparative study among them. We mention that the selection of parametric, non-parametric, and ensemble methods was based on their encompassment of techniques commonly utilized in regression problems. This study involves the creation of workloads that consume server resources and used multiple ML models to predict energy consumption using a collected database. A comparative study among different models was conducted to demonstrate the effectiveness of each model, using accuracy metrics, models complexities, and non-parametric tests.

The primary contributions of this research are detailed below:

- Creation of workloads that vary the resource utilization rates of servers in terms of CPU, RAM, and hard drive.
- Application of parametric, non-parametric, and ensemble methods on three heterogeneous servers, to verify the capability of ML models to predict the energy consumption of different servers.
- Validation of the ML models through precision measurements, non- parametric tests, and complexity.

The structure of this article is organized as follows: Section II presents the related work. Section III discusses the data center equipment and system architecture. Section IV reviews parametric, non-parametric, and ensemble methods. Section V details the data extraction processes, including the data extracted for each server, the designed algorithms, a comparison of the methods used to create workloads, and a description of the experiment. Section VI focuses on the prediction phase, applying parametric, non-parametric, and ensemble methods. Sections VII and VIII respectively present the results and discussion, including a comparison of the three types of methods across the three servers. Finally, Section IX is dedicated to the conclusion.

II. Related Work

In the literature, several authors have worked on energy management and consumption in data centers, as well as virtual machine (VM) placement on physical machines (PMs). In study [7], the authors propose two VM placement algorithms on PMs, considering CPU, RAM, memory utilization, and correlation values. These algorithms have shown performance in terms of energy consumption and service quality improvement compared to other methods [8]. The second approach involves optimal VM placement on servers using heuristics while considering hardware vulnerabilities, server energy consumption, and interference collocation among VMs [9]. This method selects an optimal approach from options such as dot product, norm2, first fit, ω-greedy, ρ-greedy, and η-greedy. Other authors in [10] studied the correlation between server resource utilization and energy consumption. They applied different workloads on two different servers and measured their energy consumption using two methods: resources utilization rate obtained from performance counters, and from external energy meter device. The results showed a strong correlation between CPU utilization and energy consumption, and a weaker correlation with RAM and disk utilization. This study can help data centers operate more energy-efficiently by optimizing resource consumption and lowering energy expenditure.

A power consumption prediction model for servers called PCP-2LSTM was proposed in study [11], this model applies a moving average smoothing technique to remove noise from the power consumption time series data and utilizes two stacked Long Short Term Memory (LSTM) networks to predict the power consumption for the next 30 seconds. A power consumption monitoring system is created to collect data and analyze power consumption, while ensuring the stationarity of the power series. CPU intensive workloads are used to collect power consumption data, and Collectd is used as a measurement tool to collect data such as CPU usage, frequency, memory usage, etc. The results show that PCP-2LSTM outperforms other models such as 2LSTM, LSTM, GBR, RAE, and ARIMA, with a normalized Root Mean Squared Error (nRMSE) of 0.417. However, it is important to note that the implementation of this model is simplified compared to a real data center, where there are various types of user requests beyond just CPU utilization. Another study proposes a VM selection policy called Maximum Correlation of sum of Squares of Deviation (MCSSD) [12]. This policy is implemented using the CloudSim tool with a heterogeneous server environment consisting of 800 servers. Workload traces from the CoMon project [13] are used, and the experiment is repeated with each server calculating resource utilization every minute. The results demonstrate that the proposed policy consumes less energy and minimizes the number of VM migrations compared to other policies such as Maximum Correlation, Minimum Utilization, and Minimum Migration Time.

Authors in study [14] developed a stochastic model to estimate the power consumption based on archived data. They consider workload and power as random variables and establish a correlation relationship between these two entities using a non-parametric approach. This makes the task complex as it requires estimating the complete distribution from the data. AI techniques were used to classify VMs based on their RAM and CPU usage [15], as well as to group user tasks based on their size and details extracted from the log file. Multiple tasks can share the same resources of a VM. The objective is to allow more dynamic resource allocation and enhance QoS standards by ensuring better resource allocation, raising user satisfaction, and lowering the number of rejected tasks. In [16], the authors present methods for evaluating and modeling the energy consumption of these resources and describe techniques that operate at the distributed system level, aiming to better manage resource scheduling, distribution, and network traffic management. Their research aims to make network and computing resources more efficient. A system proposed by [17] identifies frequently used data from application traces. Replication management and data placement are used to allocate frequently used files to "hot" disks and other files to "cold" disks. This system adds disk management to the cloud environment, which has proven effective in saving 39% of energy with an 18.26% reduction in execution time. The article also proposes an energy efficient storage system for a disk, combining energy efficient placement with a smart scheduling algorithm. The system assigns data to a disk based on its data usage model. Data replication used in the algorithm ensures fast and easy data availability. The system maximizes energy savings by associating requests to the most available disks, thereby reducing query execution time. It also considers the minimum wait time and maximum remaining idle time when research disks are active or inactive, respectively. This approach applies data replication with the appropriate number of replicas across a hybrid system, ensuring the QoS for cloud applications.

On the subject of triggering queries, researchers in study [18] noticed that the frequency of triggering the same query to access data is quite high. Therefore, forecasting and preloading often accessed queries can elevate performance levels by reducing execution time and increasing cache hit rates. Hence, they develop a prediction model that first generates memory traces to assess data usage patterns related to query frequency. Future query requests were predicted and organized using an ensemble strategy, resulting in an accuracy of 87.5%. In study [19], the authors define Random Allocation as a policy that randomly assigns arrived tasks to a queue without considering how many tasks are waiting in the queue. This can lead to overflow in one queue while others remain empty or partially filled, increasing the probability of task loss. Short tasks may wait for a long duration in front of large tasks, which the random scheduler does not detect. However, implementing this policy is easy and does not require knowledge about the system. The Shortest Queue is defined as a strategy that overcomes the problem of balanced load across queues. Upon task arrival, the policy directs them to the queues with the shortest waiting time. This ensures that full queues do not appear while others still have available capacity, reducing the probability of task loss. However, short tasks may be delayed in front of long duration tasks. This strategy only considers waiting time and not service demands. The policy requires knowledge of the queue states. A notion of Task Assignment based on Guessing Size (TAGS) is introduced, aiming to allocate tasks when the service demands are not known before execution. In this case, a task is sent to a server's queue, and the server executes the task in the queue until it is completed or the execution time elapses. In the latter case, the task is sent to another server, and the same operations are repeated, increasing the waiting time as it moves from one server to another. Compared to the shortest queue strategy and random allocation, TAGS solves the problem of short tasks waiting behind long tasks. However, performance may be affected due to repetitive service. If the waiting time is very short, multiple tasks will require repetitive services, and if it is longer, the duration will delay completion. In study [20], the authors define reliability, energy consumption, and execution time as the principal scheduling parameters for real time embedded systems. In their work, while considering three constraints: the partial order of task modules, time limitations, and reliability, based on a Directed Acyclic Graphs (DAG) and Quantum behaved Particle Swarm Optimization (QPSO). When compared to alternative algorithms, the findings demonstrate that the two proposed algorithms provide effective optimization. These two algorithms are DAG_QPSO_I and DAG_QPSO_II. The first one demonstrates efficiency in terms of energy consumption, while the second one best meets the requirements of time and reliability. In this work, workloads have been created and launched to consume resources from heterogeneous servers and ML methods belonging to different parametric, non-parametric, and ensemble methods were implemented. A comparative study was conducted between these methods for each server. Estimating server energy consumption will enable better management of energy consumption in a data center. On the other hand, data center managers can adjust resources based on server demand, which helps reduce costs. Another objective is to optimize energy efficiency by identifying energy consuming areas and implementing. In contrast to many other studies that use limited or simulated workload data, this work involves the construction and deployment of real workloads which make use of resources from diverse servers. This method provides a more accurate assessment of the energy and servers performance. Moreover, comprehensive comparisons and the selection of the optimal models for diverse circumstances are made possible by the utilization of diverse ML methods. A comparative analysis was done for each server independently to make sure the results are accurate and applicable to different servers' configurations. This degree of specificity offers useful management insights for data centers.

III. Data Center Equipment and System Architecture

The power distribution systems of the data center are intended to provide electricity to the system loads or IT and mechanical equipment, ensuring proper levels of power quality and supply security. Since the public grid may experience voltage drops or prolonged outages that can result in malfunction or even a complete shutdown of the data center, it is crucial to ensure proper power supply. In a standard data center, there is a backup diesel generator or generator set to provide power in case of major grid failures. The UPS is capable of using different storage solutions like batteries, it is typically designed to keep power supply under appropriate conditions during the startup of the diesel generator. Power supply units (PSUs) and Power distribution units (PDUs) are in charge of distributing and regulating power for servers [5]. Fig. 1. illustrate the main elements of the data center responsible for power supply. It is important to note that generators or batteries provide backup power during a power outage. Racked servers and critical datacenter infrastructure can smoothly switch to this backup power for uninterrupted service. They can support primary power during periods of high demand or grid instability. They can help to balance the load on the power infrastructure of the data center and ensure a stable and reliable supply of power to the rack servers. During peak power usage periods, when energy demand is high, generators or batteries can supplement the power supply to avoid overloading the grid. This helps manage peak demand and avoid potential blackouts. They can also provide voltage regulation to ensure that the power supplied to rack servers remains within acceptable voltage ranges. This helps maintain the stability and longevity of server hardware. Rack servers, networking equipment, storage devices and other critical infrastructure within the data center rely on stable and reliable power to function optimally. Generators and batteries provide the necessary backup power to support these loads during emergencies or planned maintenance activities. In total, effective use of generators or batteries in a data center ensures reliable power to rack servers, improves energy efficiency, and contributes to the resiliency and sustainability of the data center infrastructure.

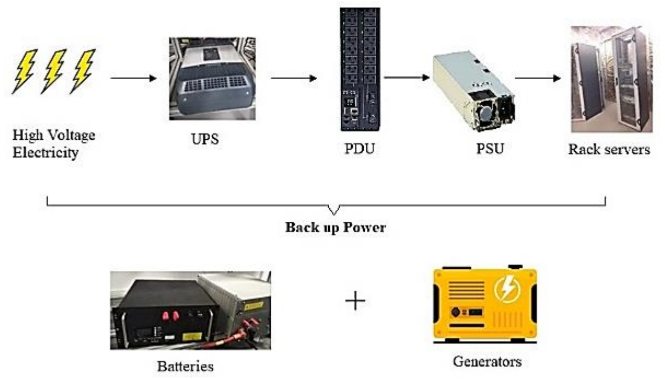

Fig. 1. Data center elements responsible for the power supply.

Most traditional data centers, including small and medium sized ones, have a power consumption breakdown as illustrated in Fig. 2 [6]. It can be observed that IT equipment accounts for 50% of the total data center power, followed by cooling systems, which represent 25% of the total power. Air handling equipment utilizes 12%, UPS units consume 10%, while lighting and other equipment consume 3% of the power. It is important to emphasize that servers, as part of the IT equipment, consume a significant fraction of the total power within data centers. This observation highlights the need to develop Machine Learning (ML) models to predict server energy consumption, enabling data center managers to better allocate their resources based on users' needs. The amount of energy consumed by servers compared to other components in a data center will vary based on factors such as workload type, server efficiency, cooling systems, and the data center design. In a conventional data center environment, as shown in Fig. 2, servers typically account for approximately 50% of total energy consumption. For this reason, and due to the limitations in accessing a real data center environment, this study focused primarily on servers, other elements such as networking equipment and cooling systems are not considered.

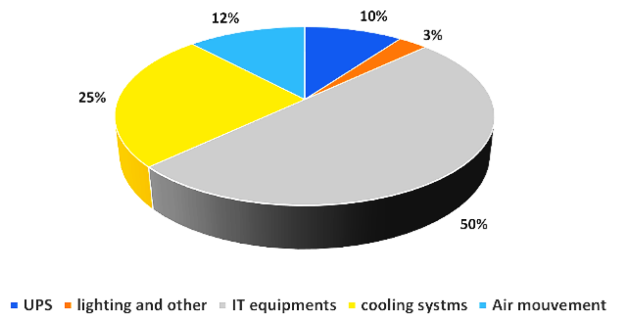

Fig. 2. Distribution of power consumption in a traditional data center [6].

Fig. 3 showcases the architecture of the system developed in this work, starting with the creation of workloads and their deployment on three servers, and ends with the generation of the proposed models.

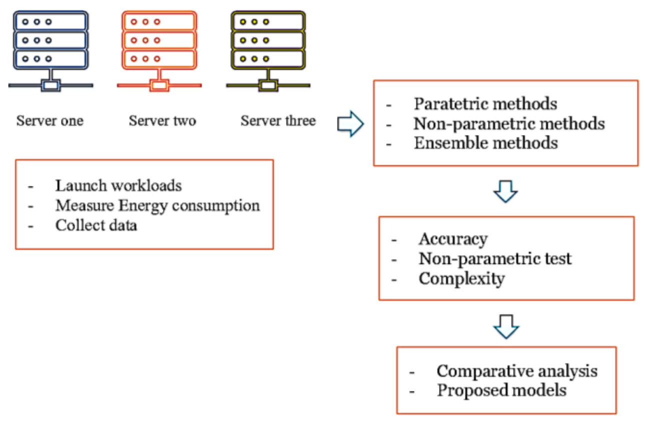

Fig. 3. System architecture.

IV. Background

A. Parametric Methods

Parametric regression techniques are statistical tools employed to examine the relationship between a dependent variable and multiple independent variables [21]. This relationship is expressed through a mathematical equation that contains a set of parameters. The objective is to estimate the values of these parameters based on the data. Common examples of parametric regression models include polynomial regression, which is a statistical study that models the variation of a dependent variable using a polynomial function of an explanatory variable. The mathematical representation of the polynomial regression is provided in Eq. (1). Where x is the independent variable, y is the dependent variable, a_i; $i=1,\dots,k$ is the coefficient or the parameter. Lasso regression (L1 regularization) is another method of statistical regression that combines linear regression with regularization, aiming to eliminate features that do not contribute to the training. Eq. (2) illustrates the formula utilized in Lasso regression, where k is the number of observations. y_i is the observed value of the observation i. x_{ij} is the value of the predictor j for the observation i. α_j is the coefficient to be estimated and strictly between zero and one. m is the number of predictor variables, and γ is the regularized parameter that controls the strength of the penalty term. Determining the ideal value of γ, or the value that strikes a balance between the model's fidelity and other factors is crucial [22]. Elastic Network (L1+L2) is a linear combination of L1 and L2 resulting in a regularizer that combines the advantages of both L1 (Lasso) and L2 (Ridge) regularization techniques, similar to Lasso regression, uses regularization to control the complexity of the model by selecting the most relevant variables. Many studies have utilized the Elastic Network, including the work presented by authors in [23], which introduces a novel algorithm for clustering analysis based on elastic networks and leveraging weighted properties. Also, in study [24] authors propose a novel approach called the Elastic Network Algorithm for Clustering based on Cluster Center Shift, which combines Mean-Shift with the Elastic Network algorithm to optimize both cluster stability and effectiveness in the cluster analysis. Lastly, Neural Networks (NN) are a type of ML that consists of interconnected neurons organized in layers and capable of learning from data by adjusting connection weights between neurons. For NN with multiple layers, the output of each layer *l* can be calculated using the output of the previous layer. Eq. (3) outlines the structural formulation of NN, where $a^{(l)}$ is the output of the layer *l*, σ is the activation function, $w^{(l)}$ is the weight matrix of layer *l*, $a^{(l-1)}$ is the input to the layer *l*, and $b^{(l)}$ is the bias for layer *l*. Lasso regression minimizes the sum of squared errors between the actual and predicted values. It works by iteratively reducing the coefficients of less important variables towards zero. It utilizes a regularization parameter, alpha, which controls the strength of the L1 penalty. A larger value of alpha leads to coefficients closer to zero. The formulation for Elastic Network can be found in Eq. (4), where *N* is the number of observations, y_i is the observed value for the observation *i*. β_k, *k=0,.., p* is the coefficient to be estimated. p is the number of predictor variables. λ_1 and λ_2 are the regularization parameters for L1 and L2 penalties. x_i is the value of the predictor variable *j* for the observation *i*. For each server, 80% of data was used for training the model, and the obtained error values were recorded. It should be noted that for each server, the models were trained using non-normalized data.

$$y = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \cdots + a_k x^k \quad (1)$$

$$\mathrm{L} = \sum_{i=1}^{k} \left(y_i - \sum_{j=1}^{m} x_{ij} . \alpha_j \right)^2 + \gamma \sum_{j=1}^{m} |\alpha_j| \quad (2)$$

$$a^{(l)} = \sigma(w^{(l)} a^{(l-1)} + b^{(l)}) \quad (3)$$

$$min_{\beta_0,\beta} \left(\sum_{i=1}^{N} (y_i - \beta_0 - x_i^T \beta)^2 + \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{j=1}^{p} \beta_j^2 \right) (4)$$

B. Non-parametric Methods

Non-parametric methods apply statistical analysis to investigate the connection between a dependent variable and multiple independent variables without making specific assumptions about the functional form or equation that describes the relationship [25]. These methods estimate the relationship based on the data itself. Common examples of non-parametric regression models include Support Vector Regression (SVR), which is a supervised learning method based on support vectors that model the data and find a hyperplane that separates the data by minimizing the squared error. It is useful for complex regression problems where the data has nonlinear relationships. The formula for SVR is presented in Eq. (5). Where ω is the weight vector or coefficients associated with the input features *x*. This vector determines the importance of each feature in the prediction. *x* is the input feature, and *b* is the bias term. Decision trees are ML algorithms used for classification and regression problems. They model the relationships between input variables and output variables using a graphical representation with connected nodes. The algorithm selects the most informative variable to split the data into categories, and the process is repeated until a stopping condition is met. Random forests are an ensemble of decision trees constructed randomly based on the training data. The prediction for an observation is based on the average or majority value of the individual predictions from the trees. Gaussian Process Regression (GPR) is a probabilistic approach based on the concept of a Gaussian process, which is a collection of random variables that have a joint Gaussian distribution for any finite number of variables. This model can capture complex and nonlinear relationships between variables without assuming a specific functional form of the relationship.

$$y = f(x) = \langle \omega, x \rangle + b; \ \omega, x \in \mathbb{R}^M, b \in \mathbb{R} \quad (5)$$

C. Ensemble Methods

Ensemble methods refer to a class of ML models that blend several models to increase the system's predictive effectiveness [26]. These models can be of the same type (homogeneous) or different types (heterogeneous). The idea behind ensemble methods is to reduce the variance and bias of individual models by combining their outputs. This is achieved through different strategies, such as bagging, which is based on an ensemble estimator using base regressors trained on random subsets of the dataset and merges their predictions, either via voting or averaging, to generate the final output. Bagging involves creating multiple versions of the same model on different subsets of the data and combining their outputs by averaging or voting. Eq. (6) illustrates the formula utilized in Bagging. Where $f_{bagging}(x)$ is the aggregated prediction for the input *x*. *N* is the total number of base learners, and $f_i(x)$ is the prediction of the base learner *i*. Boosting is a ML method that involves creating a series of models aimed at reducing errors in predictive data analysis. Each subsequent model learns from the errors of the previous one. Boosting addresses the problem of biased models when using new data by successively training multiple models to improve the overall system accuracy. Data scientists often use boosting with decision tree algorithms [27]. Boosting combines multiple models sequentially and assigns weights to the outputs of individual models, giving higher weight and input to the next model for incorrect predictions from the previous model. Stacking is an ensemble method that reduces the error rate of one or multiple estimators during prediction. It works by stacking the outputs of each individual estimator and using a regressor to compute the final prediction. This method allows leveraging the strengths of each estimator by using their outputs as input to a final estimator. It is important to mention that the base estimators are fitted on the dataset, while the final estimator is trained using the cross-validation predictions of the base estimators. Stacking incorporates boosting and aggregation, which combines the outputs of models and achieves high accuracy and low variance. The formula of Stacking can be found in Eq. (7). Where $f_{stacking}\ (x)$ is the final prediction of the stacking ensemble for input *x*, and $f_k\ (x)$, *k=1,…, N* is the predictions of base learners. *Meta_learner* is the meta learner that combines the predictions. Typically, the number of estimators used in this method ranges from three to five. In this work, parametric and non-parametric models were used and developed earlier while keeping the same chosen configurations.

$$\hat{f}_{bagging}(x) = \frac{1}{N} \sum_{i=1}^{N} \hat{f}_i(x) \quad (6)$$

$$\hat{f}_{stacking}(x) = \ Meta\_learner(\hat{f}_1(x), \hat{f}_2(x), \ldots, \hat{f}_N(x)) \quad (7)$$

V. Data Extraction

In this experiment, workloads that consume CPU, RAM, and hard disk resources of three servers were created and executed, each one is connected to a wattmeter that accurately measures the server's energy consumption with a precision of one mwh. The number of data points used for server one, server two and server three is 897, 945, and 951, respectively. Each data element represents the median value of resources utilization for the server over a period of one hour. In other words, workloads that varied the usage of CPU, RAM, and/or hard disk were executed for one hour and recorded the server's state during that execution. If the resource utilization varies frequently, the configuration is discarded; otherwise, it is kept. For the three servers, each data point (vector representing a workload) includes CPU values ranging from 10% to 100%, RAM values ranging from 10% to 90%, and disk usage ranging from zero GB to the maximum storage capacity of each server. The servers used in this study are documented in Fig. 4, which illustrates the work conducted at the LUSAC laboratory of the University of Caen Normandy in France.

Fig. 4. Servers used in the experimentation conducted at LUSAC laboratory.

In the following, the algorithms used to generate workloads that consume CPU, RAM, and disk resources of the servers are presented. For the implementation of those algorithms, the Python programming language was utilized, and every algorithm was launched in every server. After creating and executing the workloads, the energy consumption of the servers is measured. Then, on a computer, models belonging to different categories: parametric, non-parametric and ensemble, were built with the aim of predicting the energy consumption of the servers. It is mentioned that the results obtained come from a computer and not from a simulator. Also, the duration is the same in the three algorithms. For the first algorithm, OCCT was utilized to generate workloads that would stress the CPU of the server. In the second algorithm, programs such as Google Chrome were launched continuously to consume the RAM of the server, the memory usage is measured after executing those programs. In the third algorithm, files were created and duplicated on the hard drive to write data on it. After each execution, the files created to run the algorithm with different parameters were deleted (number of bytes to write). It is noted that various input data values are employed for writing operations on the hard drive. This approach aimed to create workloads that would write on the server's hard drive with continuous writing operations. Initially, the process was initiated by writing at a rate of 100 bytes per second for a duration of one hour. Subsequently, the byte values were incrementally increased in the following simulations.

```
Algorithm1: Workload_CPU
Input:   nbr_process, duration
Output:  cpu_rate, ram_rate, hard_drive_rate, energy_consumption
Start
    For var p in [1: nbr_process]:
        Create_process()
        Launch_process()
    End for
    While time < duration:
        Save (cpu_rate, ram_rate, hard_drive_rate, energy_consumption)
        If time == duration:
            For var p in [1: nbr_process]
                destroy_process ()
            End for
        End if
    End while
End
```

```
Algorithm2 Workload_RAM
Input:   nbr_process, duration
Output:  cpu_rate, ram_rate, hard_drive_rate, energy_consumption
Start
    For var p in [1: nbr_process]:
        Launch_program()
    End for
    While time < duration:
        Save (cpu_rate, ram_rate, hard_drive_rate, energy_consumption)
        If time == duration:
            For var p in [1: nbr_process]
                Destroy_process ()
            End For
            Break
        End If
    End While
End
```

```
Algorithm3: Workload_hard_drive
Input:   nbr_bytes, duration
Output:  cpu_rate, ram_rate, hard_drive_rate, energy_consumption
Start
    While time < duration:
        write_bytes()
        Save (cpu_rate, ram_rate, hard_drive_rate, energy_consumption)
        If time == duration:
            Break
        End if
    End while
End
```

The characteristics of each server are presented in Table I. It should be noted that the selection of servers is based on their availability in the research laboratory where this study is conducted.

TABLE I. SERVERS CHARACTERISTICS

Server one	Server two	Server three
processor Intel(R) Xeon(R) CPU W3565 @ 3.20 GHz, RAM 16 Go, Disk SATA SSD 256Go, NVIDIA Quadro 400, os Windows 10	Intel(R) Core (TM) i5-10500 CPU @ 3.10GHz, RAM 8GB, Disk 512GB, and OS windows server 2022 Standard version os.	Processor Intel (R) core (TM) i3 CPU 540 @ 3.07 GHz, RAM 4 Go, disk 120 Go SSD, Intel(R) HD Graphics, os Windows 10.

VI. Prediction Phase

A. Application of Parametric, Non-parametric, and Ensemble Methods on Servers

We applied parametric methods to analyze the behavior of three servers. Starting with server one, according to Fig. 5, Fig. 6 and Fig. 7 which show the error rate, the p-value, and the models' complexity respectively, It is observed that some models, such as NN, have high complexity, followed by Bagging_GPR. In terms of MSE, Lasso regression, elastic network, GPR, and other models exhibit high values around six wh. Table II presents the error rates, the p-values of non-parametric tests Wilcoxon rank sum and the complexity of each method. The results show that the polynomial regression of degree two performs the best with a low MSE of 0.71 wh. Followed by NN, which consists of two layers with 64 neurons each, using the Adam optimizer and trained with 2600 epochs. It achieves a high R² of one and low errors, indicating that this model can accurately predict the server's energy consumption. Lasso regression and Elastic Network have similar R² values, with a coefficient alpha of 0.69 for Lasso regression, an alpha and L1 coefficients of 0.34 and 0.9, respectively, for Elastic Network. The MSE of both methods exceeds five wh, indicating that these two models struggle to explain the variation in the data. The p-value of the models is relatively similar for all three models, suggesting no significant difference in their performance. For each model used, different parameters were tested, and those that yielded the best results were selected. It is interesting to note that the polynomial regression, Lasso regression, and Elastic Network models have negligible predictions and construction times. However, NN stands out with a construction time of 283 seconds, primarily due to the high number of epochs used and the complexity of its structure.

Concerning the second server, as shown in Table III, the polynomial regression of degree two had the least bias, with a MSE rate of 0.04 wh and a p-value of 0.98. Followed by NN model, configured with 2600 epochs, a sigmoid activation function for the first and the second layers, 128 neurons, and the Root Mean Squared Propagation (rmsprop) optimizer. The best result for Elastic Network was achieved with an alpha of 0.01 and an L1 of 0.01, resulting in a MSE rate of 0.07 wh. As for the Lasso regression model, it is observed that the error rate increases with the increase in the alpha coefficient. The best performance was obtained with an alpha of 0.01, resulting in a MSE of 0.08 wh. Overall, the results indicate that the parametric methods generally perform well in predicting the energy consumption for this server. Indicating a high probability that the real and predicted vectors are significantly similar. It is observed that polynomial regression, Lasso regression, and Elastic Network stand out for their reduced model construction time and negligible prediction time. On the other hand, NN model requires a longer construction time, reaching up to 110.9 s. However, it achieves a fast prediction rate of 0.1 s.

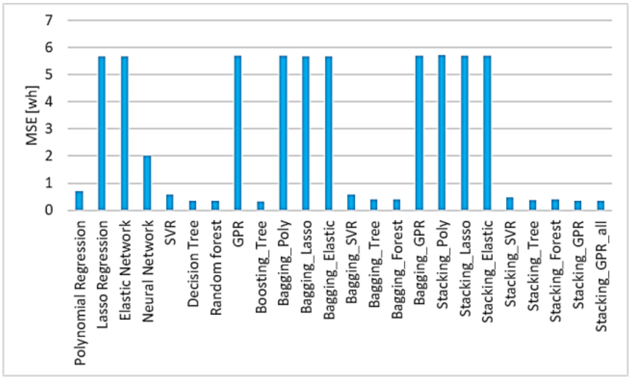

Fig. 5. MSE of server one's models.

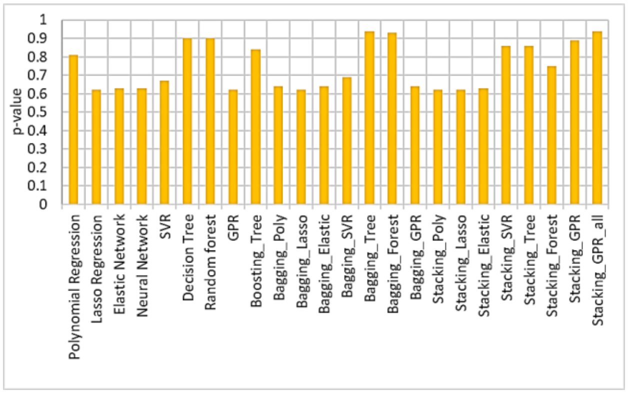

Fig. 6. P-value of server one's models.

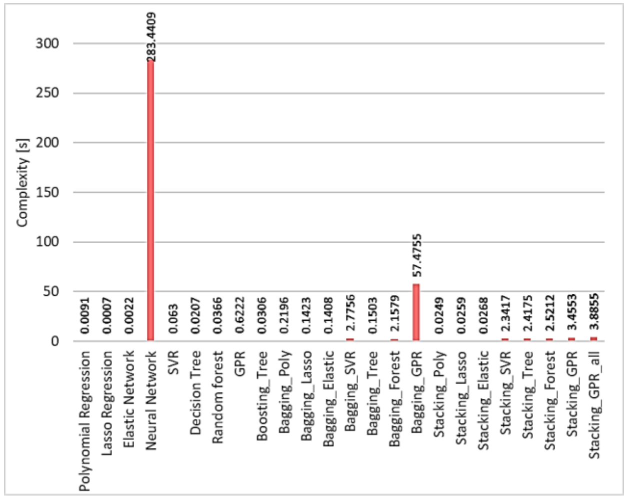

Fig. 7. Complexity of server one's models.

TABLE II. Accuracy Measurements, Non-Parametric Tests and Complexity of Parametric Methods of Server One

	R^2	MSE [wh]	MAE [wh]	RMSE [wh]	MAPE [%]	Wilcoxon rank-sum test	Model construction time [s]	Prediction time [s]
Polynomial Regression	**1.0**	**0.71**	**0.57**	**0.84**	**0.45**	**0.812**	**0.0091**	**0.0**
Lasso Regression	0.99	5.69	1.81	2.39	1.39	0.62	0.0007	0.0
Elastic Network	0.99	5.69	1.82	2.39	1.41	0.63	0.0022	0.0010
Neural Network	1.0	2.02	0.62	1.42	0.50	0.63	283.4409	0.1547

TABLE III. ACCURACY MEASUREMENTS, NON-PARAMETRIC TESTS AND COMPLEXITY OF PARAMETRIC METHODS OF SERVER TWO

	R²	MSE [wh]	MAE [wh]	RMSE [wh]	MAPE [%]	Wilcoxon rank-sum test	Model construction time [s]	Prediction time [s]
Polynomial Regression	**0.99**	**0.04**	**0.16**	**0.2**	**0.87**	**0.98**	**0.0019**	**0.0**
Lasso Regression	0.98	0.08	0.21	0.28	1.14	0.42	0.0019	0.0005
Elastic Network	0.98	0.07	0.21	0.26	1.12	0.49	0.0020	0.0
Neural Network	0.99	0.05	0.12	0.22	0.66	0.48	110.9219	0.1096

On server three, the polynomial regression model of degree two was employed, Lasso regression with an alpha of 0.01, Elastic Network with an alpha of 0.01 and an L1 of 0.01. The NN is configured with a number of epochs of 2600, two layers of 128 neurons each, using the hyperbolic tangent activation function 'tanh'. We used the Stochastic Gradient Descent optimizer 'sgd' for the NN. Table IV indicates that NN turned out to be the least biased model, with a MSE of 0.07 wh, followed by polynomial regression with a MSE of 0.09 wh. Lasso regression and Elastic Network produced similar prediction values, with a MSE of 1.24 wh and 1.23 wh, respectively. Like servers one and two, all the parametric models are characterized by their low complexity in terms of model construction and prediction. Except NN model that requires 108.34 s for model construction.

Proceeding with non-parametric methods, models were used to evaluate the performance of the servers. For the first one, the results show that decision trees, random forests, and SVR models achieve the best performance with a MSE below 0.58 wh. On the other hand, the GPR model exhibits a significantly higher MSE of 5.71 wh, indicating lower performance compared to the other models. For the SVR model, the Radial Basis Function (RBF) kernel was used with an epsilon value of 0.1 and a regularization parameter C of 19.6. The depth of the decision tree and random forests are 9 and 10, respectively, with 20 estimators for random forests. Lastly, for the GPR model, the DotProduct kernel and WhiteKernel were utilized. As observed in Table V the SVR model, decision trees, and random forests have minimal construction and prediction times, making them computationally advantageous. However, the GPR model stands out from the others in terms of construction and prediction time, being higher.

Similarly for server one, server two results, as cited in Table VI show that decision trees, random forests, and SVR are the best models for predicting server energy consumption, with a MSE rate of 0.02 wh. Although GPR model follows the other three models in terms of precision, it is also performant. We note that SVR model is configured with an epsilon and a C parameter of 0.1 and 3.0 respectively. The optimal depth for decision trees and random forests is five. The kernel used in the GPR model is based on DotProduct and WhiteKernel. Also, all selected non-parametric methods stand out for their low complexity in terms of model construction and prediction time. Likewise to the first and second servers, random forests, decision trees, and SVR are the best models for predicting the energy consumption of server three. The SVR model was trained with an epsilon of 0.06 and a C value of 16.1, while the decision trees and random forests have a maximum depth of six. On the other hand, the GPR model has a high MSE of 1.23 wh. It was trained with a DotProduct and WhiteKernel. Based on the results obtained in Table VII, it can be observed that all the proposed non-parametric models can provide energy consumption predictions with an error rate below 1.23 wh and relatively low complexity.

TABLE IV. ACCURACY MEASUREMENTS, NON-PARAMETRIC TESTS AND COMPLEXITY OF PARAMETRIC METHODS OF SERVER THREE

	R²	MSE [wh]	MAE [wh]	RMSE [wh]	MAPE [%]	Wilcoxon rank-sum test	Model construction time [s]	Prediction time [s]
Polynomial Regression	**1.0**	**0.09**	**0.21**	**0.3**	**0.41**	**0.64**	**0.0029**	**0.0**
Lasso Regression	0.98	1.24	0.88	1.11	1.81	0.35	0.0009	0.0009
Elastic Network	0.98	1.23	0.88	1.11	1.81	0.35	0.0009	0.0010
Neural Network	1.0	0.07	0.19	0.26	0.37	0.29	108.3468	0.1140

TABLE V. ACCURACY MEASUREMENTS, NON-PARAMETRIC TESTS AND COMPLEXITY OF NON-PARAMETRIC METHODS OF SERVER ONE

	R²	MSE [wh]	MAE [wh]	RMSE [wh]	MAPE [%]	Wilcoxon rank-sum test	Model construction time [s]	Prediction time [s]
SVR	1.0	0.57	0.47	0.75	0.37	0.67	0.0630	0.0079
Decision Tree	**1.0**	**0.34**	**0.36**	**0.58**	**0.3**	**0.9**	**0.0207**	**0.0**
Random forest	1.0	0.35	0.35	0.59	0.29	0.9	0.0366	0.0
GPR	0.99	5.71	1.84	2.39	1.42	0.62	0.6222	0.0315

TABLE VI. ACCURACY MEASUREMENTS, NON-PARAMETRIC TESTS AND COMPLEXITY OF NON-PARAMETRIC METHODS OF SERVER TWO

	R²	MSE [wh]	MAE [wh]	RMSE [wh]	MAPE [%]	Wilcoxon rank-sum test	Model construction time [s]	Prediction time [s]
SVR	0.99	0.02	0.1	0.14	0.53	0.9	0.0189	0.0039
Decision Tree	**1.0**	**0.02**	**0.04**	**0.14**	**0.19**	**0.9**	**0.0010**	**0.0**
Random forest	1.0	0.02	0.04	0.14	0.22	0.89	0.0927	0.0069
GPR	0.97	0.11	0.26	0.33	1.46	0.8	0.0867	0.0029

TABLE VII. ACCURACY MEASUREMENTS, NON-PARAMETRIC TESTS AND COMPLEXITY OF NON-PARAMETRIC METHODS OF SERVER THREE

	R²	**MSE [wh]**	**MAE [wh]**	**RMSE [wh]**	**MAPE [%]**	**Wilcoxon rank-sum test**	**Model construction time [s]**	**Prediction time [s]**
SVR	**1.0**	**0.06**	**0.16**	**0.24**	**0.3**	**0.86**	**0.0486**	**0.0161**
Decision Tree	1.0	0.03	0.1	0.17	0.2	0.63	0.0038	0.0
Random forest	1.0	0.02	0.09	0.14	0.19	0.48	0.1523	0.0081
GPR	0.98	1.23	0.88	1.11	1.81	0.36	0.9312	0.0081

Different ensemble methods were used, namely boosting, bagging, and stacking. The boosting model was configured with a learning rate of 0.41 and 50 estimators, using a decision tree with a depth of nine as the base estimator. The bagging model was evaluated using various regression methods as base estimators, including polynomial regression, Lasso regression, Elastic Network, SVR, decision trees, random forests, and the GPR model, denoted in Table VIII as Bagging_Poly, Bagging_Lasso, Bagging_Elastic, Bagging_SVR, Bagging_Tree, Bagging_Forest, and Bagging_GPR, respectively. The parameters for these models were defined as mentioned previously, with an optimal number of estimators set to 100. Regarding the first server, the results showed that the boosting model had the least bias, followed by the stacking model configured with GPR and the decision tree as base estimator. Next, the bagging model trained with decision trees achieved good results, as well as the stacking model configured with random forests and SVR model, along with the bagging model trained with the SVR estimator. The MSE of these models ranged from 0.32 wh to 0.59 wh, with a p-value above 0.69. However, except for the bagging model with the GPR estimator, the bagging and stacking methods trained with polynomial regression, Lasso regression, and Elastic Network as base estimators yielded less accurate prediction results compared to the aforementioned models. These models exhibited a higher MSE above five wh. It is observed that certain models had both high error rates and low model construction time, such as bagging and stacking trained with parametric methods. The bagging using the GPR model is distinguished with high error rates and significant model construction time. On the other hand, other models such as boosting and bagging using non-parametric methods other than GPR, as well as stacking using non-parametric models or a combination of the two methods, were characterized by low error rates and model construction times below 3.8 s.

Moving to the second server, the MSE of the boosting model is 0.02 wh and the R² reaches one, indicating that the model is capable of predicting energy consumption with negligeable error. As shown in Table IX, the p-value of the Wilcoxon tests is indicating that this method provides predicted values similar to the real values. By applying bagging and stacking models with different estimators. The best results were obtained using decision trees, random forests, and polynomial regression as base estimators for both methods. The average MSE in these models is below 0.07 wh, with a high value of R². The stacking model trained with the GPR estimator is the least biased in terms of precision measurement among all the proposed ensemble methods, with a MSE of 0.01 wh. The Bagging_Lasso, Bagging_Elastic, Bagging_GPR, Stacking_Lasso, and Stacking_Elastic models follow the previously mentioned models in terms of prediction quality. Among the ensemble methods, it is noteworthy that all models are characterized by their low complexity, with a model construction time of 9.18 s for the bagging model trained with the base estimator GPR.

TABLE VIII. ACCURACY MEASUREMENTS, NON-PARAMETRIC TESTS AND COMPLEXITY OF ENSEMBLE METHODS OF SERVER ONE

	R²	**MSE [wh]**	**MAE [wh]**	**RMSE [wh]**	**MAPE [%]**	**Wilcoxon rank-sum test**	**Model construction time [s]**	**Prediction time [s]**
Boosting_Tree	**1.0**	**0.32**	**0.36**	**0.57**	**0.29**	**0.84**	**0.0306**	**0.0**
Bagging_Poly	0.99	5.7	1.83	2.39	1.42	0.64	0.2196	0.0159
Bagging_Lasso	0.99	5.69	1.81	2.39	1.39	0.62	0.1423	0.0059
Bagging_Elastic	0.99	5.69	1.82	2.39	1.40	0.64	0.1408	0.0049
Bagging_SVR	1.0	0.59	0.48	0.77	0.37	0.69	2.7756	0.8097
Bagging_Tree	1.0	0.4	0.36	0.63	0.29	0.94	0.1503	0.0069
Bagging_Forest	1.0	0.41	0.37	0.64	0.3	0.93	2.1579	0.1466
Bagging_GPR	0.99	5.7	1.84	2.39	1.42	0.64	57.4755	0.0947
Stacking_Poly	0.99	5.74	1.84	2.4	1.43	0.62	0.0249	0.0009
Stacking_Lasso	0.99	5.7	1.83	2.39	1.41	0.62	0.0259	0.0009
Stacking_Elastic	0.99	5.7	1.83	2.39	1.42	0.63	0.0268	0.0009
Stacking_SVR	1.0	0.47	0.4	0.69	0.31	0.86	2.3417	0.0269
Stacking_Tree	1.0	0.37	0.37	0.61	0.3	0.86	2.4175	0.0119
Stacking_Forest	1.0	0.41	0.4	0.64	0.32	0.75	2.5212	0.0139
Stacking_GPR	1.0	0.34	0.36	0.58	0.29	0.89	3.4553	0.0119
Stacking_GPR_all	1.0	0.35	0.36	0.59	0.29	0.94	3.8855	0.0209

TABLE IX. ACCURACY MEASUREMENTS, NON-PARAMETRIC TESTS AND COMPLEXITY OF ENSEMBLE METHODS OF SERVER TWO

	R²	MSE [wh]	MAE [wh]	RMSE [wh]	MAPE [%]	Wilcoxon rank-sum test	Model constr-uction time [s]	Prediction time [s]
Boosting_Tree	1.0	0.02	0.04	0.14	0.24	0.82	0.4188	0.0
Bagging_Poly	0.98	0.07	0.21	0.26	1.13	0.45	0.1134	0.0059
Bagging_Lasso	0.97	0.11	0.27	0.33	1.48	0.6	0.1276	0.0049
Bagging_Elastic	0.97	0.11	0.26	0.33	1.46	0.59	0.1386	0.0069
Bagging_SVR	0.99	0.02	0.09	0.14	0.49	0.92	0.6927	0.3383
Bagging_Tree	**1.0**	**0.02**	**0.04**	**0.14**	**0.19**	**0.87**	**0.1269**	**0.0069**
Bagging_Forest	1.0	0.02	0.01	0.14	0.2	0.87	2.2184	0.1476
Bagging_GPR	0.97	0.11	0.26	0.33	1.46	0.82	9.1893	0.2563
Stacking_Poly	0.98	0.07	0.21	0.26	1.13	0.46	0.0469	0.0
Stacking_Lasso	0.96	0.16	0.35	0.4	1.92	0.36	0.0399	0.0
Stacking_Elastic	0.98	0.1	0.25	0.32	1.41	0.31	0.0411	0.0
Stacking_SVR	1.0	0.02	0.07	0.14	0.37	0.46	0.8902	0.0089
Stacking_Tree	0.99	0.03	0.04	0.17	0.24	0.87	0.9402	0.0079
Stacking_Forest	1.0	0.02	0.04	0.14	0.22	0.79	0.9835	0.0099
Stacking_GPR	1.0	0.01	0.05	0.1	0.26	0.49	1.8894	0.0109
Stacking_GPR_all	1.0	0.02	0.05	0.14	0.25	0.88	1.9824	0.0134

Lastly, as for the third server, it can be observed that the boosting method trained with decision trees of depth six as base estimators provides predicted values with a MSE of 0.03 wh. Similarly, the bagging and stacking methods trained with the SVR model, decision trees, and random forests as base estimators have a MSE ranging from 0.02 wh to 0.06 wh. Therefore, the stacking model trained with GPR has an MSE of 0.03 wh. On the other hand, the other models such as Bagging_Poly, Bagging_Lasso, Bagging_Elastic, Bagging_GPR, Stacking_Poly, Stacking_Lasso, and Stacking_Elastic provide prediction values with an MSE of 1.23 wh and 1.24 wh. It can be observed that all ensemble methods are capable of predicting energy consumption with an error rate below 1.24 wh in terms of MSE, MAE, and MAPE, and p-values ranging from 0.35 to 0.89, as indicated in Table X. The values of the MAPE are already multiplied by 100. Therefore, a MAPE of 0.45 represents 0.45% and not 45%. Different complexity characteristics are observed among the boosting and bagging models using parametric models as base estimators. They have relatively low complexity, making them efficient in terms of model construction time. Next, bagging using decision trees, random forests, and SVR as base estimators, which also exhibit moderate but slightly higher complexity. On the other hand, the bagging model using GPR as the base estimator stands out for its higher model construction time, reaching 93.19 s. As for stacking, the results show that the methods using parametric models have a model construction time of 1.1 s, indicating relatively low complexity. On the other hand, stacking using non-parametric models has model construction times ranging from 5.3 s to 8.08 s, which remains reasonable considering the more complex nature of these models.

TABLE X. ACCURACY MEASUREMENTS, NON-PARAMETRIC TESTS AND COMPLEXITY OF ENSEMBLE METHODS OF SERVER THREE

	R²	MSE [wh]	MAE [wh]	RMSE [wh]	MAPE [%]	Wilcoxon rank-sum test	Model construction time [s]	Prediction time [s]
Boosting_Tree	**1.0**	**0.03**	**0.1**	**0.17**	**0.21**	**0.89**	**0.0637**	**0.2169**
Bagging_Poly	0.98	1.24	0.88	1.11	1.81	0.37	0.1585	0.0100
Bagging_Lasso	0.98	1.24	0.88	1.11	1.81	0.36	0.1902	0.0080
Bagging_Elastic	0.98	1.24	0.88	1.11	1.81	0.37	0.1912	0.0080
Bagging_SVR	1.0	0.06	0.16	0.24	0.31	0.9	6.2082	0.9499
Bagging_Tree	1.0	0.02	0.09	0.14	0.19	0.48	1.3107	0.0080
Bagging_Forest	1.0	0.03	0.1	0.17	0.2	0.45	3.8057	0.1723
Bagging_GPR	0.98	1.24	0.88	1.11	1.81	0.38	93.1952	0.3245
Stacking_Poly	0.98	1.24	0.88	1.11	1.8	0.38	1.1771	0.0
Stacking_Lasso	0.98	1.23	0.88	1.11	1.81	0.35	1.1857	0.0020
Stacking_Elastic	0.98	1.23	0.88	1.11	1.81	0.35	1.1792	0.0
Stacking_SVR	1.0	0.03	0.1	0.17	0.2	0.38	5.8112	0.0201
Stacking_Tree	1.0	0.02	0.09	0.14	0.19	0.67	5.3006	0.0182
Stacking_Forest	1.0	0.05	0.1	0.22	0.22	0.53	5.6406	0.2202
Stacking_GPR	1.0	0.03	0.11	0.17	0.21	0.68	7.6102	0.0302
Stacking_GPR_all	1.0	0.03	0.11	0.17	0.23	0.65	8.0803	0.0202

VII. Results

This section provides an overview of the results obtained. For the first server, observations indicate that certain parametric methods, specifically polynomial regression, perform well in terms of prediction with low error rates and reduced complexity. However, other methods like Lasso regression and Elastic Network show high error rates exceeding five wh in terms of MSE for energy consumption prediction, while maintaining relatively low complexity. Among the non-parametric methods, SVR, decision trees, and random forests have low complexity and error rates for energy consumption prediction. However, GPR model provides prediction results with a high error rate and higher complexity compared to other methods. When it comes to ensemble methods, the boosting method trained with decision trees provides results very close to the actual values with low complexity. It is to highlight that models trained with parametric models have high MSE error rates but low complexity. Conversely, models trained with non-parametric methods like SVR, decision trees, and random forests as base estimators are considered the best models with construction complexity ranging from 0.15 s to 2.77 s. However, using the GPR base estimator in the bagging method significantly increases the MSE error rate up to 5.7 wh, and the model construction complexity reaches 57.47 s.

When using the stacking method, all models trained with parametric methods provide poor prediction results. On the other hand, models trained with non-parametric methods show optimal error rates. In conclusion, the recommended model for predicting the energy consumption of this server is the stacking model, using non-parametric or both parametric + non-parametric methods as estimators and trained with GPR as final estimator. The advantage of this method is that it combines multiple ML models and selects the optimal model to provide the best results. Furthermore, the complexity of this method is low. For example, although the GPR model provides poor prediction results when used alone, it can be used as the final estimator in the stacking model to improve the model's error rate and obtain optimal results.

For server two, it is remarkable that all models have a MSE lower than 0.17 wh. Among the parametric methods, polynomial regression and NN are characterized by their high performance. However, it should be noted that NN has considerable complexity due to their architecture and computational operations involved. Next, Lasso regression and Elastic Network, which exhibit average error rates and lower complexity. For non-parametric methods, the SVR model, decision trees, and random forests achieve a MSE of 0.02 wh, indicating good prediction performance. However, the GPR model has a higher error rate, suggesting poorer performance compared to other models. In terms of complexity, all methods are characterized by low complexity, making them efficient in terms of model construction and prediction time. The boosting method offers optimal prediction results with low complexity, as behave the bagging methods trained with non-parametric models such as SVR, decision trees, and random forests, as well as the stacking methods trained with non-parametric models. The recommended model for predicting energy consumption while maintaining a trade-off between complexity and error rate is the stacking model trained with non-parametric or mixed models. Regarding complexity, all proposed methods have low complexity, except the bagging model using the base estimator GPR, which has a model construction complexity of 9.18 s.

It is observed for the third server that high performing parametric methods with low error rates are polynomial regression and NN. However, the other two parametric methods reach a MSE of 1.23 wh, with a p-value of 0.35. It is important to note that for all three servers, NN are considered complex due to their architecture. Therefore, if new data is added to retrain the model, it may not be the optimal choice in terms of complexity, even though its performance may surpass some other models. As with server one and server two, non-parametric methods other than GPR provide optimal prediction, and the complexity of all methods is low. For ensemble methods, bagging_SVR, bagging_Tree, bagging_Forest, stacking_SVR, stacking_Tree, stacking_Forest, and stacking_GPR are considered optimal in terms of MSE, MAE, and MAPE. However, some models have low error rates while others have higher error rates, notably the stacking method using non-parametric methods as estimators, resulting in complexity reaching 7.61 s. Similar to server one and server two, bagging using GPR as base estimator stands out with a higher error rate and higher complexity of 93.19 s. Other models such as bagging and stacking trained with parametric models have a MSE error rate of 1.24 wh and low complexity. The best model for predicting energy consumption of this server while maintaining a balance between complexity and error rate is the Gradient Boosting model. The stacking method is characterized by a low error rate. However, the model construction complexity is higher compared to the Gradient Boosting model, reaching up to eight seconds. These observations allow us to better understand the relationship between different models and their respective complexity. They also highlight the importance of considering this complexity when choosing and evaluating models to find the right balance between performance and computational efficiency.

VIII. Discussion

The variation in servers’ energy consumption with workload execution is illustrated in Fig. 8, when CPU_workloads are executed on the first server, the CPU utilization rate reaches 94.5%, resulting in an increase in server energy consumption from 98.52 wh in idle state to 177.92 wh under load. On the other hand, when RAM_workloads are executed, energy consumption reaches 130.93 wh, while CPU utilization varies between 11.2% and 24.9%. Finally, when disk-based workloads are executed and write up to 11 290.35 MB on the disk, energy consumption reaches 125.84 wh, with RAM utilization varying between 24.33% and 71.17%, and CPU utilization varying between 12.5% and 24.2%. For server two, energy consumption ranges from 16 wh to 22.73 wh when CPU-intensive workloads are executed, with CPU utilization varying between 8.1% and 92.4%. When RAM-intensive workloads are executed, energy consumption reaches 20.32 wh, with RAM utilization varying from 26.38% to 78.29%. For disk-intensive workloads, energy consumption ranges from 16.99 wh to 18.75 wh, with CPU and RAM utilization rates ranging from 10.7% to 16.6% and 51.1% to 78.29% respectively, and data written to the disk reaching up to 18 064.42 MB. Regarding server three, energy consumption varies from 41.66 wh to 62.4 wh when executing workloads that consume between 25.6% and 100% of the CPU. For workloads that vary RAM utilization between 40.12% and 68.21% and CPU utilization between 22.2% and 23.1%, energy consumption ranges from 40.77 wh to 42.25 wh. Launching workloads that consume the hard disk results in energy consumption ranging from 35.72 wh to 44.91 wh, with CPU utilization varying between 24.6% and 33.9%, and RAM utilization ranging from 42.74% to 49.56%, with data written to the disk reaching up to 4 393.99 MB.

Note that for all three servers, each parameter was carefully selected through experimentation. Experiments were conducted extensive testing on a range of values, choosing those that yielded the best results in terms of accuracy, complexity, and non-parametric test performance. Additionally, the same workloads were launched on the three servers, but each server handled them according to its architecture and capacity. For example, when a workload is launched on server *i*, it may complete successfully within the expected duration, while on another server, it may crash due to insufficient capacity to process and execute that workload. This explains why the number of data varies from one server to another.

The MSE, complexity, and p-value performance of the three servers are shown in Fig. 9. For the three servers, it was identified that the best models were polynomial regression, decision tree, boosting_Tree, and bagging_Tree. The outcomes demonstrate that the three servers' MSEs are all less than 0.7 wh, their complexities are all under 1.31 seconds, and their p-values range from 0.48 to 0.98. These findings show those models are useful for low computational complexity and high accuracy server energy consumption prediction. Because the models' predictions are statistically similar to the actual values, as confirmed by the high p-values, they can be effectively implemented to improve energy management and efficiency.

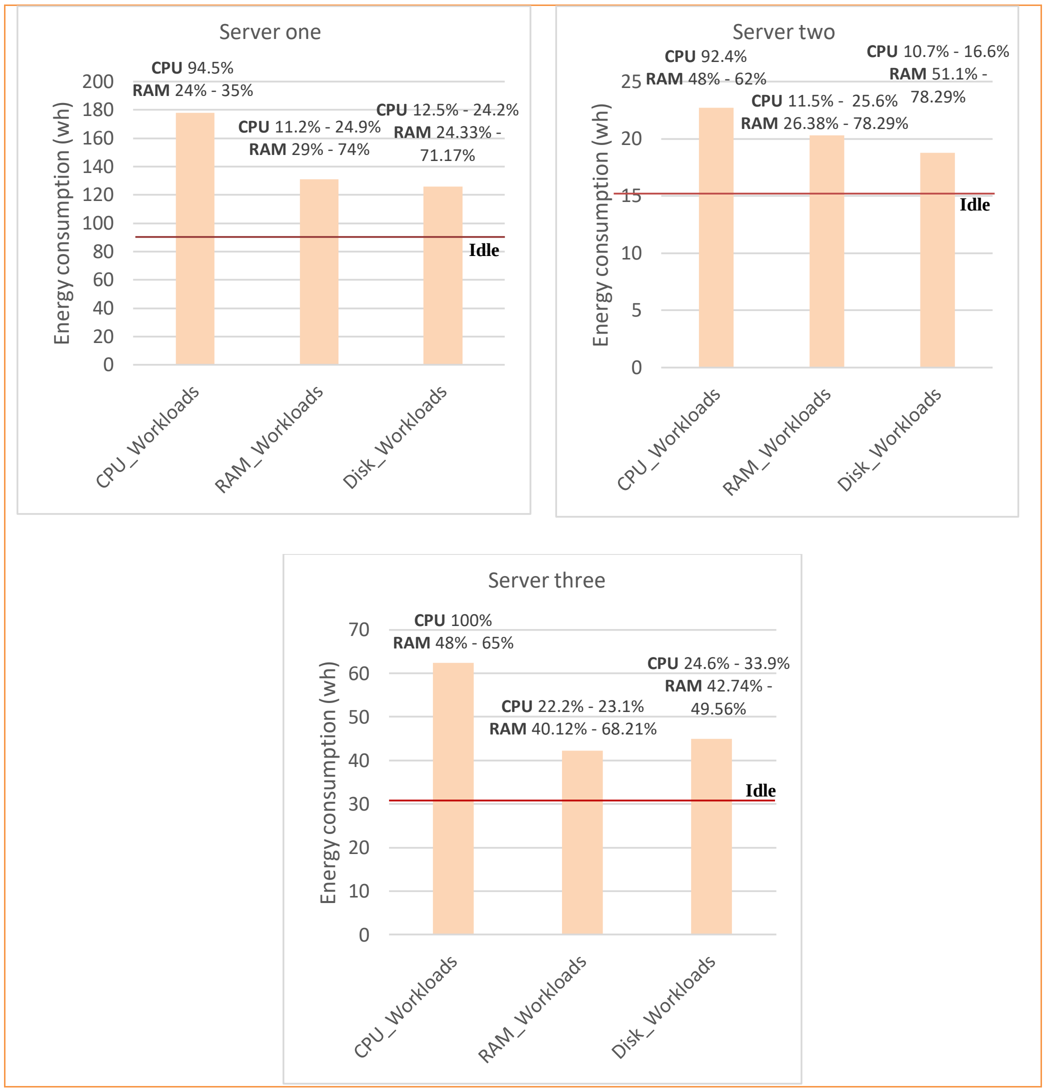

Fig. 8. Variation in servers energy consumption with workload execution.

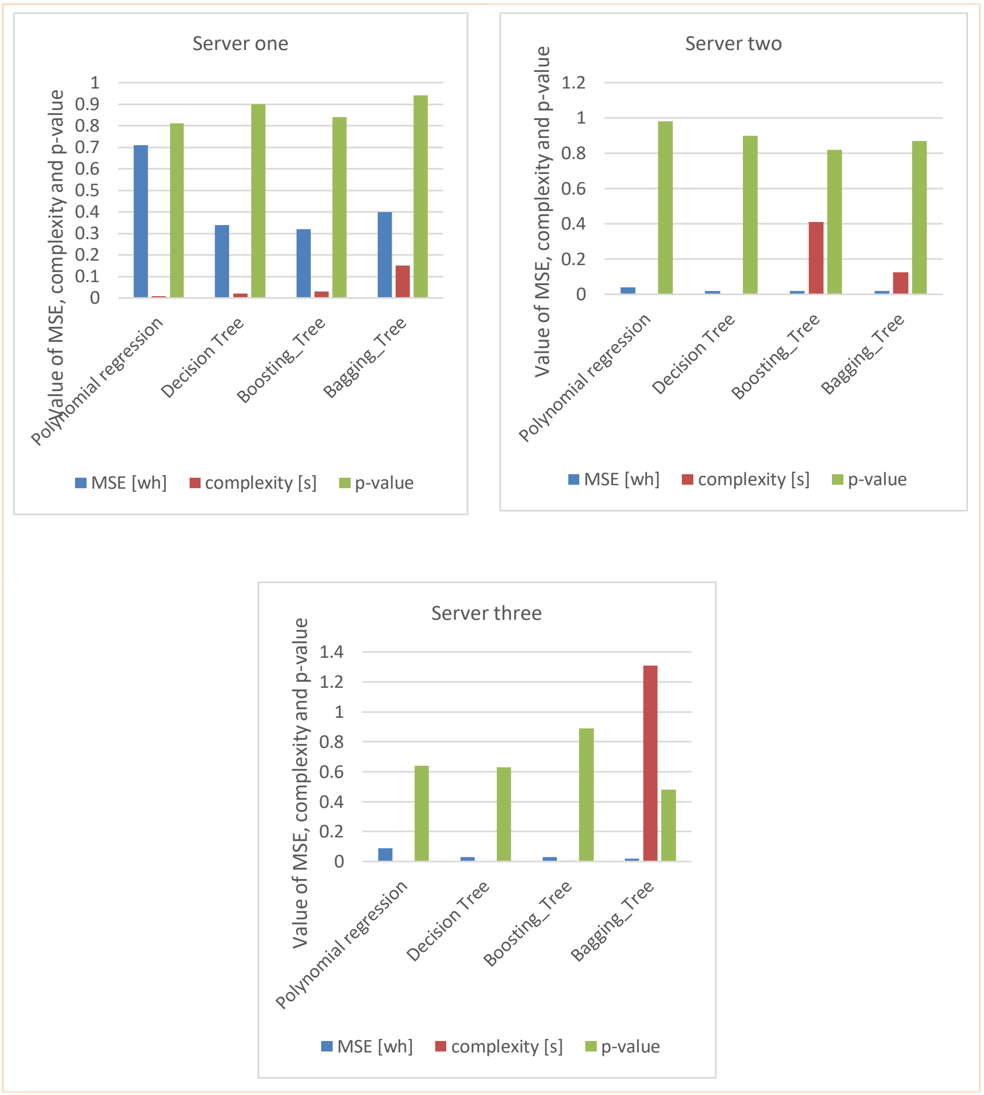

Fig. 9. Performance of the top models for the three servers.

Observation shows that for the three servers, certain methods are distinguished by low error rates and low complexity. These methods include polynomial regression belonging to parametric methods, as well as SVR, decision trees, and random forests belonging to non-parametric methods. Similarly, Boosting and Stacking with parametric models show good performance in terms of error rates. Other methods, such as Lasso regression, Elastic Network, GPR, and bagging using parametric models, have high error rates but low complexity. On the other hand, bagging methods using SVR, decision trees, and random forests, as well as Stacking with non-parametric models, are characterized by low error rates and moderate complexity. The worst results in terms of high error rates and high complexity are obtained with NN and bagging using GPR as the base estimator. Comparing the three servers, it is observed that each has a different energy consumption pattern. Server one consumes 2.85 times more energy than server three and 7.82 times more than server two. Server three, on the other hand, consumes 2.74 times more energy than server two. In this study, the complexity of the model was included as a critical factor for several reasons. For instance, after selecting a model that meets the requirements, it becomes crucial to assess its complexity. If the data center providers intend to retrain the model with additional data, understanding its complexity becomes paramount. If the complexity is high, retraining the model might not be advisable. However, if the complexity is manageable, this process can enhance the model's capability to accommodate new workloads, thereby improving its estimation accuracy. It should be highlighted that, to apply the elaborated algorithms to a real data center, it is highly recommended to use the data directly from the servers within that data center. In fact, the variability in workload characteristics influences the creation of ML models designed to predict server energy consumption. In this work, three types of servers were used to demonstrate the performance of ML models, as the type of servers may vary from one data center to another. This study aims to pinpoint the most effective models for estimating server energy usage, offering tailored recommendations for different server environments.

Predicting the future workload of servers in a data center is considered a challenging task because it is difficult to control or fully anticipate the clients' needs, as they are free to use or consume resources according to the SLA. However, by analyzing historical server activity, patterns can be identified using various ML methods to estimate server workloads. In this study, the focus is placed on resources such as the CPU, RAM, and hard disk. Although users may request other types of resources, this study is limited to the resources. If the data center workloads shift in an irregular way and atypical manner, predictions can still be made by the proposed ML models. Besides, in this situation, it will be advised to use continuous learning, which will enable ML models to automatically update with current data and adjust over time to shifting workload patterns. Additionally, retraining the ML models on current data on a regular basis will keep them more accurate and relevant. Additionally, online learning has also the potential to be effective and enable the models to rapidly adapt to transient variations in workload.

IX. Conclusion

The choice of the appropriate method depends on the desired objectives. If the focus is on prediction quality in terms of error rates, the Stacking model, which uses the GPR model alongside other parametric and non-parametric models as internal estimators, is a good choice for predicting server energy consumption. However, if the objective is to regularly retrain the model by increasing the amount of data, it is preferable to choose a model that has both low error rates and reduced complexity. In this case, the polynomial regression model may be a good choice due to its architecture and computational requirements. It can be observed that for all three servers, the CPU consumes the most energy compared to the other resources, and the energy consumption when applying workloads that consume RAM and hard disk differs from one server to another. In this study, only CPU, RAM, and hard drive workloads are used as data to construct ML models to predict energy consumed by servers, while other types of workloads can also be handled within the data center. Also, in real data center, servers are placed in racks and installed in rooms equipped with other elements such as cooling system. However, the integration of cooling units and heat management are not elaborated in this work. Our future research will focus on exploring and refining anomaly detection in data centers as a critical frontier for ensuring the robustness and reliability of server systems by enabling the identification and mitigation of irregularities or unexpected patterns in energy consumption. Ultimately, the efforts in anomaly detection promise not only to improve system reliability, but also to facilitate more sustainable and resource-efficient operation of server networks.

Ethical Approval

As this research does not involve human participants or animals, ethical approval was not required for this study.

Competing interests

The research conducted in this article is primarily intended for academic and educational purposes.

Authors' contributions

Meryeme EL YADARI created the database, performed data analysis, conducted the experiments, and wrote the article.

Saloua EL MOTAKI contributed to this work by proposing ideas, reviewing the article, enhancing its content, and correcting the report.

Ali YAHYAOUY, and Khalid EL FAZAZY collaborated in this work by organizing meetings, where we discussed the work, provided valuable feedback, and made suggestions for content improvement.

Hamid GUALOUS provided the servers and all necessary materials for conducting the experiments, participated in the meetings, validating the work, and offering valuable remarks.

Stéphane LE MASSON actively engaged in discussions with the authors, providing insightful feedback to enhance the content of the work.

Funding

All the materials used in the experiment were provided by the University laboratory of applied sciences of Cherbourg (LUSAC).

Availability of Data and Materials

The database utilized in this work is internally developed and considered confidential; thus, there is no external link available for accessing it. The materials utilized in the experiment consist of three servers responsible for executing the tasks, one computer dedicated to data manipulation, and three wattmeters for measuring energy consumption.

References

[1] Robertazzi, Thomas. 2012. Data Centers. In Basics of Computer Networking, ed. Thomas Robertazzi, 69–72. SpringerBriefs in Electrical and Computer Engineering. New York, NY: Springer.

[2] Wang, Ting, Yu Xia, Jogesh Muppala, and Mounir Hamdi. 2018. Achieving Energy Efficiency in Data Centers Using an Artificial Intelligence Abstraction Model. IEEE Transactions on Cloud Computing 6: 612–624.

[3] Hasan, Zuhair. 2009. Redundancy for data centers. ASHRAE Journal 51. American Society of Heating, Refrigerating, and Air-Conditioning Engineers, Inc. (ASHRAE): 52–55.

[4] Lakshmi, Ganesh. 2012. Data center energy management. Faculty of the Graduate School of Cornell University: Cornell.

[5] Oró, Eduard, Victor Depoorter, Albert Garcia, and Jaume Salom. 2015. Energy efficiency and renewable energy integration in data centres. Strategies and modelling review. Renewable and Sustainable Energy Reviews 42: 429–445.

[6] Li, Munan, and Alan Porter. 2019. Can nanogenerators contribute to the global greening data centres? Nano Energy 60. https://doi.org/10.1016/j.nanoen.2019.03.046.

[7] El Yadari, Meryeme, Ali Yahyaouy, Khalid El Fazazy, Stéphane Le Masson, and Hamid Gualous. 2022. Placement methods of Virtual Machines in servers. In 2022 International Conference on Intelligent Systems and Computer Vision (ISCV), 1–7. Fez, Morocco. https://doi.org/10.1109/ISCV54655.2022.9806069.

[8] Baskaran, Nithiya, and Eswari R. 2021. Efficient VM Selection Strategies in Cloud Datacenter Using Fuzzy Soft Set. Journal of Organizational and End User Computing (JOEUC) 33. IGI Global: 153–179.

[9] Caviglione, Luca, Mauro Gaggero, Massimo Paolucci, and Roberto Ronco. 2021. Deep reinforcement learning for multi-objective placement of virtual machines in cloud datacenters. Soft Computing 25: 12569–12588.

[10] El Yadari Meryeme, Ali Yahyaouy, Stéphane Le Masson, Khalid El Fazazy, and Hamid Gualous. 2022. Study of the correlation between

server resources utilization and energy consumption. In , 6. Marseille, France: IEEE. https://doi.org/10.1109/ICSC57768.2022.9993950.

[11] Shen, Ziyu, Xusheng Zhang, Binghui Liu, Bin Xia, Zheng Liu, Yun Li, and Saiqin Long. 2019. PCP-2LSTM: Two Stacked LSTM-Based Prediction Model for Power Consumption in Data Centers. In 2019 Seventh International Conference on Advanced Cloud and Big Data (CBD), 13–18. Suzhou, China: IEEE.

[12] Reddi, Kamal Sandeeep, and Syam Kumar Pasupuleti. 2019. Optimal Energy aware Dynamic Virtual Machine consolidation in Cloud Data Centers. In 2019 IEEE 16th India Council International Conference (INDICON), 1–4. Rajkot, India: IEEE.

[13] Park, KyoungSoo, and Vivek S. Pai. 2006. CoMon: a mostly-scalable monitoring system for PlanetLab. ACM SIGOPS Operating Systems Review 40: 65–74.

[14] El Motaki Saloua, Ali Yahyaouy, and Hamid Gualous. 2022. Modeling the correlation between the workload and the power consumed by a server using stochastic and non-parametric approaches, June 13, John Wiley&Sons, Ltd edition.

[15] Elrotub, Mousa, and Abdelouahed Gherbi. 2018. Virtual Machine Classification-based Approach to Enhanced Workload Balancing for Cloud Computing Applications. Procedia Computer Science 130: 683–688.

[16] Orgerie, Anne-Cecile, Marcos Dias de Assuncao, and Laurent Lefevre. 2014. A survey on techniques for improving the energy efficiency of large-scale distributed systems. ACM Computing Surveys 46: 1–31.

[17] Arora, Sumedha, and Anju Bala. 2021. An intelligent energy efficient storage system for cloud based big data applications. Simulation Modelling Practice and Theory 108: 102260.

[18] Arora, Sumedha, and Anju Bala. 2021. An ensembled data frequency prediction based framework for fast processing using hybrid cache optimization. Journal of Ambient Intelligence and Humanized Computing 12: 285–301. https://doi.org/10.1007/s12652-020-01973-5.

[19] Alssaiari, Ali, and Nigel Thomas. 2020. Energy Consumption by Servers under Unknown Service Demand. Electronic Notes in Theoretical Computer Science 353: 21–38.

[20] Xiong, Wei, Bing Guo, and Shen Yan. 2022. Energy consumption optimization of processor scheduling for real-time embedded systems under the constraints of sequential relationship and reliability. Alexandria Engineering Journal 61: 73–80.

[21] Yavuz, Esra, and Mustafa Şahin. 2022. Investigation of Parametric, Non-Parametric and Semiparametric Methods in Regression Analysis. Sakarya University Journal of Science.

[22] Lakshmi, Mayur V., and Joab R. Winkler. 2024. Numerical properties of solutions of LASSO regression. Applied Numerical Mathematics. https://doi.org/10.1016/j.apnum.2024.03.010.

[23] Liu, Jingang, Xiaofeng Cao, Yilong Lei, and Jiayuan Zhang. 2023. The Application of Elastic Network Algorithm Based on Weighted Property in Clustering Analysis. IEEE Access 11: 136077–136090. https://doi.org/10.1109/ACCESS.2023.3335139.

[24] Junyan, Yi, and Du Xiaopeng. 2020. Elastic Network Algorithm for Clustering Based on Cluster Center Shift. In 2020 Chinese Control And Decision Conference (CCDC), 1971–1976. https://doi.org/10.1109/CCDC49329.2020.9164435.

[25] CFI, Team. 2020. Nonparametric Method. Corporate Finance Institute. June 30.

[26] Lutins, Evan. 2017. Ensemble Methods in Machine Learning: What are They and Why Use Them?

[27] AWS. 2023. Qu'est-ce que le boosting ? – Le boosting dans le cadre du machine learning expliqué – AWS. Amazon Web Services.