Energy Consumption Analysis of Predictive Modeling Optimization with Hyperparameter Tuning

Aida Nauanova
Master of Science in Information & Technology - Sustainable Computing
a.nauanova2892@student.leedsbeckett.ac.uk
Supervisor: Professor Ah-Lian Kor

***Abstract*—Machine Learning models are increasingly used in many realms, often without due consideration for their computational footprint, including energy consumption, which can cause a hazardous impact on the environment. This study investigates the performance of various machine learning models in predicting energy consumption while considering the environmental impact of the models themselves. Three regression models – XGBoost, Random Forest Regression, and Linear Regression – are employed for energy consumption prediction. GridSearchCV is used for hyperparameter tuning of the linear regression and XGBoost models, while Random Forest Regression is additionally optimized with a combination of RandomSearchCV and Bayesian Optimization. The energy consumption associated with training and running each model is measured using the CodeCarbon library after 10 training iterations. An analysis of variance (ANOVA) test is then conducted to determine statistically significant differences in the energy consumption of the models themselves. This research contributes to the growing field of environmentally aware machine learning by exploring the trade-off between model prediction accuracy and the associated carbon footprint.**

I. Introduction

A. Background of Context

The rising global energy consumption demands exploration of sustainable solutions in many realms. Buildings are significant contributors to energy usage, accounting for a substantial portion of total energy demand [1]. Effective building energy management hinges on a comprehensive understanding of the factors that influence energy consumption patterns. These factors can be broadly categorized into three groups: building characteristics, occupant behavior, and external conditions.

By analyzing historical data on energy consumption, weather conditions, occupancy patterns, and building characteristics, ML models can learn relationships and predict future energy usage. This information empowers building managers to implement targeted energy-saving strategies. However, training and deploying ML models also has an environmental cost. The computational resources required for training can contribute to carbon emissions. It leads to another concern that should not neglected, to consider both the model's effectiveness in predicting energy consumption and its environmental footprint throughout the development and usage lifecycle.

This study explores the application of machine learning for building energy prediction, focusing on prediction accuracy by employing hyperparameter tuning, and environmental impact by estimating the carbon footprint of models. This will give an opportunity to build even more effective energy management systems. The findings can guide future research towards developing more environmentally friendly ML approaches for building management applications.

B. Aim and Research Objectives/Questions

This research addresses building energy management through two key aspects. First, it focuses on developing accurate forecasting models for building energy consumption and optimizing them with hyperparameter tuning. Second, the research investigates the environmental impact associated with the models themselves. By utilizing codecarbon, the energy consumption during the inference stage of the optimized models will be measured. The following research objectives are established to achieve the mentioned goal:

1) Explore the dataset
2) Build various machine learning models
3) Evaluate the performance of predictive models
4) Improve models using hyperparameter tuning
5) Measure the carbon footprint of models' performance
6) Complete a statistical analysis of results

C. Rationale

The rationale of this research lies in impact to the effective and environmentally friendly energy management. Building energy efficiency improvements can reduce energy use by 30 to 80 percent and the accompanying building carbon emissions significantly [2]. It further aims to explain the importance of using certain methods of hyperparameter tuning that cause the least amount of emissions.

D. Contribution of Research

This study will test the hypothesis of using hyperparameter tuning to optimize each model's performance to see whether it improves or not. A comprehensive statistical analysis will be conducted using CodeCarbon to assess the carbon footprint associated with training and deploying each optimized model. This combined analysis aims to identify models that achieve a good balance between prediction accuracy and environmental cost. The research results will enable building managers to make informed decisions regarding model selection and deployment. Statistical analysis from the CodeCarbon can be used to establish thresholds for acceptable environmental impact, recommending models that balance this with reliable forecasts.

II. Literature Review

Machine learning (ML) has revolutionized various disciplines, but its growing complexity raises concerns about its environmental impact. Training and deploying ML models can consume significant energy, leading to a carbon footprint.In their article titled "How to Estimate and Reduce the Carbon Footprint of Machine Learning Models," Groes and Ludvigsen (2023) address this critical issue [3]. They acknowledge the growing body of research highlighting the environmental cost of ML development and propose an approach to mitigate this impact [3]. Their work contributes to ongoing efforts to establish more sustainable practices within the field of machine learning. Recent research emphasizes the substantial energy consumption, particularly large language models (LLMs) and deep learning architectures [4], [5]. Strubell et al. [4] quantify the energy demands of training LLMs and states the need for energy-efficient practices. Similarly, Patterson et al. [5] discuss the growing energy consumption of cloud-based ML training. These studies underscore the importance of understanding and mitigating the environmental impact of ML models.

The importance of artificial intelligence (AI) in smart cities for collecting information and distribution is the main topic of Bibhu and Pawankumar's research (2022) [6](diploma). The writers go through a number of AI applications, such as sentiment analysis and natural language processing. They also emphasize the potential advantages of AI, such as enhanced communication and more effective execution of public services. A machine learning-based strategy for comprehending and tailoring smart city services utilizing the Internet of Things and large data was proposed in one of the articles of 2017 [7]. Users were grouped according to their likes and interests using a clustering technique, and customized services were then offered using predictive models. A thorough analysis of the uses of artificial intelligence and machine learning in smart cities was carried out [8]. A number of application cases, including those involving traffic control, energy conservation, and public safety were discovered.

Hashem I. et al. in his research discuss the role of big data in smart cities [9]. The authors comprehensively review existing literature on smart cities, big data analytics, and their interrelationship. Additionally, the review covers the challenges associated with big data and smart cities, such as privacy, security, and ethical concerns. The article "Machine learning based system for managing energy efficiency of public sector as an approach towards smart cities" by Zekić-Sušac, Mitrovic, and Has (2021) presents a comprehensive study on the use of machine learning to manage energy efficiency in the public sector [10]. Based on the literature review findings, the authors proposed a system for controlling the efficiency of energy in the public sector. The system consists of four main components: data acquisition, data prepossessing, machine learning modeling, and energy management decision-making.

The article by Sathishkumar V E., Shin, C., & Cho, Y. (2021) presents an efficient prediction model for an industry building in a smart city that is optimized using hyperparameter tuning [11]. The study uses a quantitative research methodology, where the authors collected data from an industrial building in South Korea for one year, from January to December 2018. The authors used statistical techniques such as correlation, regression, and principal component analysis to identify the key variables influencing energy consumption. They developed a predictive model using artificial neural networks (ANN), then compared its performance to other models. The study results indicated that the ANN-based optimized model outperformed the other models regarding accuracy and reliability.

III. Methodology

A. Macro Methodology

This research leverages an empirical evaluation macro-methodology, which prioritizes acquiring knowledge through experimentation and subsequent data analysis. The study employs an experimental approach to assess the energy consumption and accuracy of three machine learning models: linear regression, random forest regression, and XGBoost, optimized using hyperparameter tuning. By analyzing the collected data on energy consumption, which was generated using a CodeCarbon, the research aims to identify which models achieve the best balance between two critical factors; to provide valuable insights for developing more sustainable real-world machine learning applications, where both performance and environmental impact are crucial considerations.

B. Infrastructure Architecture

1) Hardware: The experiments conducted on a personal computer equipped with an AMD Ryzen 5 3550H processor with 8 cores, 5.88 GB of available RAM, and a dual-GPU configuration consisting of an NVIDIA GeForce GTX 1650 and integrated Radeon Vega Mobile Gfx. This configuration provides sufficient computational resources for training and evaluating the chosen machine learning models (linear regression, random forest regression, and XGBoost).

2) Software:

- Jupyter Notebook
  This web-based interactive development environment was used for data exploration, model development, training, and initial evaluation.
- Visual Studio Code
  This code editor was used for writing and executing Python scripts specifically designed for hyperparameter tuning and integration with CodeCarbon for carbon footprint estimation.
- Google Colaboratory
  Google Colab notebook was used for a statistical analysis of results generated by CodeCarbon to define the significance of difference using the ANOVA test.

C. Micro Methodology

Three models were trained for this experiment and three optimization methods were used. Linear regression and XGBoost models were optimized using the Grid Search Cross-Validation technique, and Random Forest was optimized using three techniques: GridSearchCV, RandomSearchCV and Bayesian optimization.

1) Dataset: The dataset used for experiments in this study was taken from an open-source Kaggle. This dataset contains various features relevant to building energy consumption, such as temperature, humidity, occupancy, and details on heating, ventilation, air conditioning (HVAC), and lighting usage, as well as energy consumption measurements. Additionally, it includes information on renewable energy contributions. The environment in which measurements were taken is stimulated and created for research purposes. This gives an opportunity to analyze energy consumption patterns in great detail and build models to predict future usage. The dataset is meticulously crafted to mirror real-world conditions, capturing the subtle variations that influence energy use [12].

2) Grid Search Cross-Validation: The GridSearch technique is used to find the optimal combination of hyperparameter values for the linear regression, random forest regression, and XGBoost model. It was employed during the model development phase to tune the performance of a model. In a grid search, a predefined set of hyperparameter values was defined for each hyperparameter of the model. The technique exhaustively searched through all possible combinations of these values to identify the combination that yields the best performance according to a chosen evaluation metric [13].

3) RandomSearchCV: This study additionally employed RandomSearchCV for hyperparameter tuning of the random forest regression model. Similar to GridSearchCV, RandomSearchCV explores a predefined grid of hyperparameter values. However, it departs from the exhaustive evaluation employed by GridSearchCV. Instead, RandomSearchCV randomly samples a user-specified number of hyperparameter configurations from the defined grid [14].

4) Bayesian Optimization: Bayesian Optimization utilized a probabilistic model to iteratively select the most promising hyperparameter configurations for evaluation. This model is constructed based on the performance of previously evaluated configurations.

$$\text{EI(x)} = \text{E}\,[\max(0, f(\mathbf{x}) - \mu(\mathbf{x}))],$$
$$\text{UCB(x)} = \mu(\mathbf{x}) + \kappa\sqrt{\sigma^2(\mathbf{x})},$$

Mathematically, a Bayesian acquisition function (Expected Improvement (EI) or Upper Confidence Bound (UCB)) is used to balance exploration and focus on high-performing regions. This iterative process efficiently allocates computational resources towards configurations with the highest potential for improvement, making it particularly suitable for models with expensive evaluation costs [15].

5) Extreme Gradient Boosting model: XGBoost algorithm that combines the principles of gradient boosting with decision trees to create accurate and powerful regression models. XGBoost regression was chosen for these experiments as it optimizes a loss function by adding weak decision trees to an ensemble every iteration [16]. GridSearchCV was chosen as the initial hyperparameter tuning technique for the XGBoost model. This approach provided a comprehensive exploration of the defined hyperparameter grid, establishing a baseline performance for the model.

6) Linear Regression: Due to the smaller number of hyperparameters involved, GridSearchCV was implemented to optimize the linear regression. An exhaustive search which is computationally efficient for linear regression models, avoids evaluating a set of combinations.

7) Random Forest Regression: Similar to linear regression, GridSearchCV provides a structured and interpretable approach to hyperparameter tuning. You can easily see how different hyperparameter combinations affect model performance, aiding in understanding the model's behavior [4].

The following evaluation metrics were used to improve the performance of prediction models:

- Mean Absolute Error
- Mean Squared Error
- Root Mean Squared Error
- R-squared score

IV. Findings and Discussion

Fig. 1 provides insights into the trend of energy consumption of the dataset. The blue line shows the hourly consumption of the energy and the red line is the weakly rolling mean.

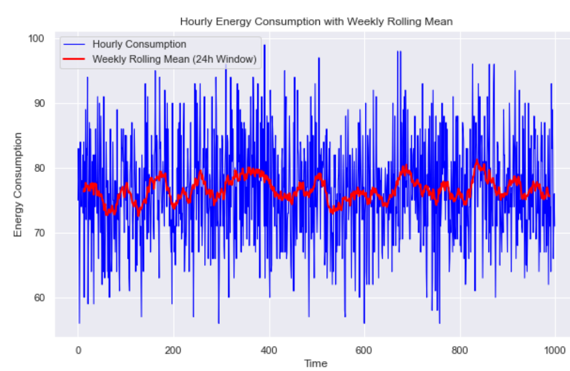

Fig. 1. Hourly energy consumption with weekly rolling mean

A. Prediction Accuracy

The dataset was divided into training and testing sets using an 80/20 split. The training set (80%) was used to train the models, and the testing set (20%) was used for evaluation. The performance of each model was evaluated using metrics stated before. The table below represents the accuracy before hyperparameter optimization.

Model	R2 score	MSE	RMSE	MAE
Linear regression	0.62	26.76	5.17	4.17
RF Regression	0.55	31.55	5.62	4.67
XGBoost	0.53	32.68	5.72	4.62

Hyperparameter optimization techniques were employed to improve the performance of the models. As can be seen from the table below, there are slight improvements in the model R2 score.

Model	R2 score	MSE	RMSE	MAE
LR with GridSearchCV	0.62	26.83	5.18	4.18
RF with GridSearchCV	0.56	30.71	5.54	4.51
RF with Bayesian	0.59	28.82	5.37	4.40
RF with RandomizedSearchCV	0.58	29.49	5.43	4.48
XGBoost with GridSearchCV	0.60	27.86	5.28	4.36

B. Energy Consumption Measurements

The previous work was exported in .py format to continue the experiment in the Visual Studio Code platform. Using the Python package CodeCarbon, it was possible to track the emission produced during the model training and inference. CodeCarbon package provides an opportunity to capture measurements into certain files. This way 5 CSV format files were produced, each of 10 rows as a result of 10 iterations with various metrics including:

- **timestamp:** The date and time at which the training run commenced.
- **project_name:** The identifier assigned to the specific model training project.
- **run_id:** A unique identifier for each individual training run within a project.
- **duration (seconds):** The total time taken for the model training run to complete.
- **emissions (kg CO_2):** The estimated amount of carbon dioxide equivalent (CO_2) emissions generated during the training run. CO_2 is a metric that accounts for the combined impact of various greenhouse gases.
- **emissions_rate (kg CO_2/second):** The average rate of COe emissions produced per second of training.
- **cpu_power (W):** The average power consumption (in watts) of the central processing unit (CPU) during training.
- **gpu_power (W):** The average power consumption (in watts) of the graphics processing unit (GPU) if utilized during training (many models don't require a GPU).
- **ram_power (W):** The average power consumption (in watts) of the random access memory (RAM) during training.
- **cpu_energy (kWh):** The estimated energy consumption (in kilowatt-hours) by the CPU during training (calculated from duration and CPU power).
- **gpu_energy (kWh):** The estimated energy consumption (in kilowatt-hours) by the GPU during training (if used, calculated from duration and GPU power).
- **ram_energy (kWh):** The estimated energy consumption (in kilowatt-hours) by the RAM during training (calculated from duration and RAM power).
- **energy_consumed (kWh):** The total estimated energy consumption (in kilowatt-hours) during the training run (sum of CPU, GPU, and RAM energy).
- **country_name:** The name of the country where the training run was executed.
- **country_iso_code:** The two-letter ISO code for the country where the training run was executed.

By analyzing these metrics, the environmental impact of different model training configurations can be identified. Additionally, Codecarbon provides information about the hardware that was used during the model training. This way it can be used for research purposes to track the differences of emissions on different machines. Fig.2 presents the energy consumption of each model and tuning technique, where:

- LR - Linear Regression
- RF - Random Forest regression

It is seen from Fig.2 that Random Forest regression that is optimized with GridSearchCV, consumes the biggest amount of energy per training iteration. Other optimization techniques used for Random Forest regression consume less energy, but still more than compared to Linear Regression and XGBoost models.

C. Statistical Analysis

An Analysis of Variance (ANOVA) test was implemented to statistically analyze the carbon footprint estimates generated by CodeCarbon. To define the significance of the difference between models, it was decided to choose the "energy_consumed" column. Using the measurements of 5 different optimization techniques' energy consumption, the ANOVA test showed the following results:

```
F Value = 103.868
P Value = 4.01424e-22
The null hypothesis can be rejected
```

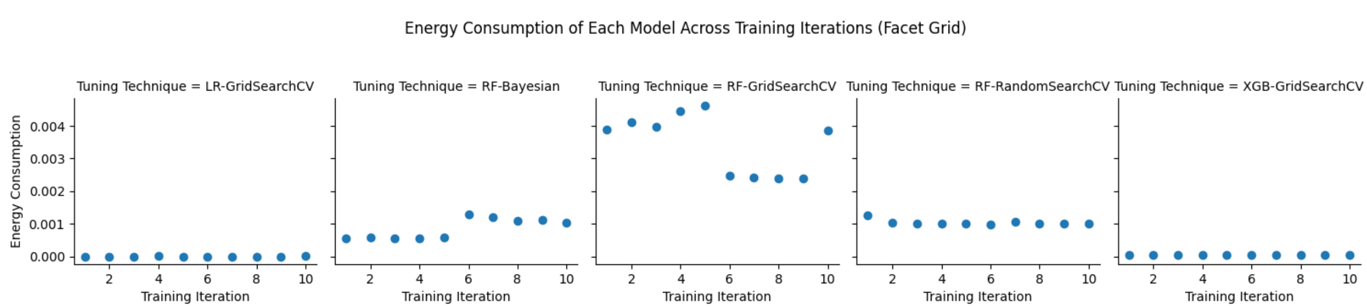

Fig. 2. Energy Consumption of each model

The "emissions" column was also statistically analyzed to define whether there is a significant difference in the amount of emissions produced by model training. The results state that the p-value is less than the alpha value which is 0.01, which means that there is a significant difference.

```
F Value = 104.026
P Value = 3.8926e-22
The null hypothesis can be rejected
```

V. Conclusion and Recommendations

References

[1] U.S. Energy Information Administration (EIA), "Electricity," https://www.eia.gov/electricity/, U.S. Energy Information Administration (EIA).

[2] O. R. J. K. S. Z. Ian H., Harry K., "Global status report for buildings and construction," 2020. [Online]. Available: https://globalabc.org/resources/publications/2020-global-status-report-buildings-and-construction

[3] K. Groes and A. Ludvigsen, "How to estimate and reduce the carbon footprint of machine learning models," https://medium.com/@biksharma17/machine-learning-model-to-predict-and-reduce-carbon-dioxide-emissions-192a2828c3dc, 2023, towards Data Science.

[4] E. S. et al., "Energy and policy considerations for deep learning in nlp," 2019.

[5] D. P. et al., "Machine learning at scale and in the cloud," *ACM Queue*, vol. 11, no. 4, pp. 10–35, 2013.

[6] P. S. Bibhu D., "Role of artificial intelligence in smart cities for information gathering and dissemination (a review)," 2022. [Online]. Available: https://doi.org/10.52132/Ajrsp.e.2022.39.4

[7] I. L. Jeannette C., Vic C., "Understanding and personalising smart city services using machine learning, the internet-of-things and big data," 2017. [Online]. Available: https://doi.org/10.1109/ISIE.2017.8001570

[8] L. M. R. G. Zaib U., Fadi A., "Applications of artificial intelligence and machine learning in smart cities," 2020. [Online]. Available: https://doi.org/10.1016/j.comcom.2020.02.069

[9] C. V. A. N. B. A. K. S. Y. I. G. A. A. E. C. H. Hashem, I. A. T., "The role of big data in smart city. international journal of information management, 36(5), 748–758." 2016. [Online]. Available: https://doi.org/10.1016/j.ijinfomgt.2016.05.002

[10] M. S. H. A. Zekić-Sušac, M., "Machine learning based system for managing energy efficiency of public sector as an approach towards smart cities. international journal of information management, 58, 102074." 2021. [Online]. Available: https://doi.org/10.1016/j.ijinfomgt.2020.102074

[11] C. C. Y. Sathishkumar V E., Shin, "Efficient energy consumption prediction model for a data analytic-enabled industry building in a smart city. building research and information, 49(1), 127–143." 2021. [Online]. Available: https://doi.org/10.1080/09613218.2020.1809983

[12] K. Datasets, "Energy Consumption Prediction Dataset," https://www.kaggle.com/datasets/mrsimple07/energy-consumption-prediction/data, [Accessed: 2024-04-30].

[13] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, and J. Vanderplas, *Scikit-learn: Machine Learning in Python*. Jupyter Notebook, 2011.

[14] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, no. Feb, pp. 281–305, 2012.

[15] J. Snoek, H. Larochelle, and R. Adams, "Practical bayesian optimization of machine learning algorithms," in *Proceedings of the 25th International Conference on Neural Information Processing Systems*, vol. 1, 2012, pp. 255–263.

[16] XGBoost Documentation Team, "Xgboost: A scalable machine learning system for gradient boosting," https://xgboost.readthedocs.io/.