21st International Conference Man and Working Environment

SEMSIE

Safety Engineering & Management - Science, Industry, Education

ORIGINAL SCIENTIFIC PAPER

DOI: 10.46793/SEMSIE25.087M

MIRJANA MILUTINOVIĆ[1]
MLADENA LUKIĆ[2]

[1,2]University of Niš,
Faculty of Occupational Safety, Serbia

[1]*mirjana.milutinovic@znrfak.ni.ac.rs*
[2]*mladena.lukic@znrfak.ni.ac.rs*

HOW GREEN IS YOUR ALGORITHM? ASSESSING THE CARBON FOOTPRINT OF MACHINE LEARNING

Abstract*: The rapid advancement of machine learning (ML) and its extensive application across various fields have led to numerous innovative uses. However, large-scale ML systems require significant computational resources, energy usage, and result in associated carbon emissions, which have raised some concerns. ML can combat climate change with smart decision-making, but energy-intensive models like deep learning also impact the environment. In this paper, we applied the CodeCarbon, an open-source tool for estimating energy consumed and carbon dioxide (CO_2) emissions during the runs of ML models (Linear Regression, k-Nearest Neighbors Regressor, and Decision Tree Regressor). Both default and optimized models show low CO_2 emissions, with optimization resulting in slightly higher values. The impact of geographical locations related to the carbon intensity of electricity generation on emissions is also examined, along with the effects of utilizing the complimentary cloud service GoogleColab. Due to low emissions, applied ML algorithms are suitable for education, research, and practice. The increasing use of artificial intelligence (AI) highlights tracking of carbon emissions, even in lightweight ML algorithms, to introduce sustainable AI practices. The aim of this paper is to raise awareness of the energy and environmental cost of AI at all levels of research.*

Keywords: sustainable AI, carbon footprint, machine learning models, CodeCarbon, GoogleColab

ORCID iDs: Mirjana Milutinović ● N/A
Mladena Lukić ● https://orcid.org/0000-0003-1105-3637

INTRODUCTION

Over recent decades, the increasing occurrence of extreme weather events (storms, droughts, wildfires, and floods) has underscored the destructive consequences of climate change. Global emissions of greenhouse gases (GHG) have continued to increase over the past several decades, despite widespread scientific agreement on the importance of reducing these emissions. Comprehensive response to tackle climate change involves both mitigation and adaptation strategies. Recent rapid scientific and technological developments have increased the application of artificial intelligence (AI) in various sectors to address climate change, including energy, transportation, industry, land use, and disaster resilience (Kaack, 2021). Machine learning (ML), a branch of AI, is increasingly used in predicting, classifying, approximating, and optimizing climate datasets for real-time monitoring, trend forecasting, and risk estimation (Olawade, 2024a). Although AI is widely recognized as an appropriate tool for achieving sustainability and net-zero emissions (Olawade, 2024b; Rolnick et al., 2019; Vinuesa, 2020), there is evidence that its role is twofold. ML training and a vast number of parameters tuning (as in deep learning and large language models) can be time and energy demanding. The iterative training and hyperparameter tuning of ML architectures are computationally intensive, leading to increased electricity consumption and higher GHG emissions (Strubell 2020). Many scientists suggest shifting the focus from achieving high accuracy to achieving high efficiency, due to the extensive use of AI with significant computational costs and large carbon footprints (Schwartz, 2020; Bergstra, 2011). According to the Intergovernmental Panel on Climate Change (IPCC) projections, limiting carbon emissions to 2 metric tons of CO_2 [1]equivalent per person per year is necessary to maintain the global temperature rise below 1.5°C (IPCC, 2023). Emissions of 100 megatonnes of CO_2 per year produced by data centers and high-performance computing (HPC) facilities could have a major impact on climate change in the future (Lannelongue 2021, Luccioni, 2023; Strubell, 2019, Lannelongue, 2023, Bannour, 2021, Dodge, 2022, Henderson, 2020). Energy consumption is linked

[1] The CO_2 equivalent or CO_2eq is a metric measure used to compare the emissions from various GHG based on their Global Warming Potential (GWP), by converting amounts of other gases to the equivalent amount of CO_2 with the same GWP.

to various activities, including the production of computer hardware, model development, data storage, data transfer, data processing, and cooling of large data centres. According to research, NVIDIA's new AI servers are projected to consume more energy than Argentina and Sweden by 2027 (Dayarathna, 2015; Xing, 2023). Since improvements in hardware, software and algorithms have significantly enhanced scientific research, there is an estimation of 9 billion hours spent on scientific computing in 2020, but the computational cost of such research activities and contribution of GHG emissions has not been completely explored (Extreme Science and Engineering Discovery Environment XSEDE). Also, to assess the AI environmental impact, it is important to know what kind of electricity is generated in a country − renewable or non-renewable. The uneven distribution of renewable and non-renewable electricity sources worldwide results in a significant AI carbon footprint for some countries (Electricity Maps). This raises a critical question: are we considering AI, on its current path, as sustainable? Concerns about the potentially huge negative environmental impact of AI in the future have established a new sustainable approach−green AI. As the name suggests, Green AI promotes performance/efficiency trade-off, reduction of computational cost, and reasonable use of resources (renewable energy sources) (Alzoubi, 2024; Schwartz 2020). Various tools can estimate GHG emissions during machine learning runs, including Green-Algorithms, CodeCarbon, Eco2AI, CarbonTracker, MLCO2, and Cumulator. Various studies summarise different methods for measuring energy consumption and emissions associated with computing activities (Bouza et al.). 2023, Lannelongue 2023).

This paper utilizes CodeCarbon to estimate the carbon footprint during the execution of three machine learning algorithms: Linear Regression, k-Nearest Neighbors Regressor, and Decision Tree Regressor. CodeCarbon is an open-source tool designed to estimate the amount of CO_2 produced by the cloud or personal computing resources used to execute the code (CodeCarbon). Since energy efficiency and carbon emissions have become a critical issue in ML research and deployment, our aim was to compare emissions from both default and optimized ML algorithms and support sustainable, energy-efficient AI practices, particularly in education, by tracking CO_2 emissions. The European Federation of Academies of Sciences and Humanities reports that the emission of CO_2 equivalents per scientist per year ranges from 1 to 30 tonnes (ALLEA 2022), accounting for emissions from computing, laboratories (LEAF framework), supply chains, conferences, and other related activities. Although ML algorithms commonly used in education (by students, academics, researchers, practitioners, etc.) have a low and moderate complexity (unlike deep learning and large language models with billions of parameters), the impact they may have in the future due to increasing use should be investigated. Although the results indicate that CO_2 emissions from some machine learning algorithms are significantly lower than those generated by GPT training (552 tCO_2eq) (Patterson, 2021), our objective is to raise awareness of the energy and environmental costs associated with AI tools at all levels of research. However, most ML applications in industry and education rely on simpler models like Linear Regression and Decision Trees. Their carbon footprint is often considered negligible and ignored. This paper aims to fill that gap by quantifying and analysing emissions from lightweight models using the CodeCarbon library.

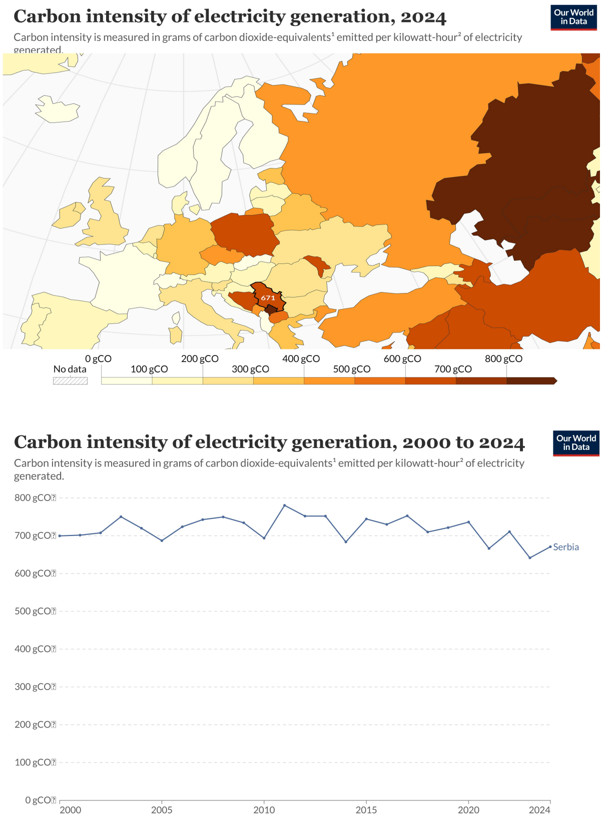
METHODOLOGY

CodeCarbon is an open-source library simply implemented in Python code. This tool tracks the emission of CO_2 during the ML run. Emission of CO_2 is the product of the carbon intensity of electricity generation (a given country) and energy consumed. Carbon intensity is a known amount of CO_2 emitted for each kilowatt-hour (kWh) of electricity generated, calculated as a weighted average of the emissions from the different energy sources fossil fuels (coal, petroleum, natural gas) and renewable (solar power, hydroelectricity, biomass, geothermal, wind, nuclear, etc.) (Electricity Maps). Carbon intensity is measured as kilograms (or grams) of CO2 emitted per kilowatt-hour of electricity (kg/kWh). Energy consumed (quantified as kWh) is the amount of electrical energy used by the computational infrastructure (machine) during the execution of the code.

$$\textit{CO}_2\textit{eq emissions (kg) = carbon intensity (kg/kWh) X energy consumed (kWh)}$$

CodeCarbon uses global carbon intensity of electricity per country, representing average value from the previous year. Electricity in Serbia is primarily generated from non-renewable sources, leading to a carbon intensity of 0.671 kgCO2eq/KWh. This is higher compared to Norway and Switzerland, which have carbon intensities of 0.031 kgCO2eq/KWh and 0.037 kgCO2eq/KWh, respectively (Ember). Carbon intensity of electricity generation defines emissions, and for the same energy consumption ML application in Serbia produce more than 20 times higher emission than in Norway and Switzerland. Figure 1a) shows global carbon intensity of electricity generation in Europe for 2024, and 1b) annually carbon intensity values for Serbia during the period from 2000 to 2024. The values are shown in grams (g).

CodeCarbon measures energy consumption from three main hardware parts: Graphics Processing Unit (GPU), Central Processing Unit (CPU) and memory. Energy consumption by CPU can be estimated using two methods. The first method estimates energy consumption multiplying training duration by thermal design power – TDP, representing the maximum amount of power that CPU or GPU can dissipate when operating at maximum performance. The second method uses software integrated tool Running Average Power Limit (RAPL) which monitors and controls the power usage of the processor and its components (Bouza, 2023). Efficiency of the computing facility or Power Usage Efficiency (PUE) is the efficiency coefficient of the data centre. Recommended PUE value for personal computers is 1. In this experiments ML training and testing are performed on the same hardware setup to ensure comparability. Training carried out with an Intel Core i5 processor (CPU), integrated GPU and 16 GB RAM (Table 1).


Carbon intensity of electricity generation, 2024

Carbon intensity is measured in grams of carbon dioxide-equivalents[1] emitted per kilowatt-hour[2] of electricity generated.

Carbon intensity of electricity generation, 2000 to 2024

Carbon intensity is measured in grams of carbon dioxide-equivalents[1] emitted per kilowatt-hour[2] of electricity generated.

Figure 1. a) *Carbon intensity of electricity generation per country (average values for 2024). Carbon intensity for Serbia is 671 gCO2.eq/KWh. b) Carbon intensity for Serbia during the period from 2000 to 2024. Source: Ember (2025); Energy Institute - Statistical Review of World Energy (2024).*

Default and optimized version of three ML algorithms (Linear Regression, k-Nearest Neighbors Regressor, Decision Tree Regressor) is evaluated by CodeCarbon to assess CO$_2$ emission. In a regression model, the output as a dependent variable is obtained as a function of the independent input variable (Alpaydın, 2010). ML training is performed using supervised learning which is based on a dataset of input and corresponding output data. Training set consists of pairs of input and associated output data. ML algorithms reach a defined performance level when the difference between the actual output and the anticipated output becomes minimal.

Linear Regression is an easy, understandable, interpretable and applicable model with low computational complexity and fast training. Linear regression assumes a linear relationship between variables and usually does not perform well with complex, non-linear data. The fitting line is obtained using the least squares method (which minimizes the difference between the anticipated and the actual values). In an optimized linear regression model, generalization is enhanced, and overfitting is reduced, allowing minimal energy consumption.

The *k-Nearest Neighbors Regressor* is a non-parametric model, applicable for complex relationships. It is adaptable to different data types and easy easy-to-implement model. k-Nearest Neighbors predicts a value of new data based on its similarity to nearest data in the training set, e.g., assumes that nearest objects are similar (Altman, 1992). Its performance varies significantly with data size and dimensionality, and with large datasets algorithm can be computationally expensive. The optimization focuses on dimensionality reduction and efficient searching, which reduces inference time and memory usage, leading to lower energy demand of the optimized model.

Decision Tree Regressors uses a tree-like structure to make predictions by splitting data into subsets. A decision tree is composed of internal decision nodes and terminal leaves. A decision tree predicts responses to data by following the decisions in the tree from the root (beginning) down to a leaf node. A tree consists of branching conditions where the value of a predictor is compared to a trained weight. During the training number of branches and the values of weights are set (Alpaydın, 2010). Decision Tree Regressors can predict continuous values of variables and deal with non-linear relationships, but it is prone to overfitting and requires carefully selected data. Optimizations such as reducing, limiting maximum tree depth, and setting minimum sample thresholds per node effectively control model complexity, reduce training time, and enhance interpretability, contributing to a measurable decrease in energy usage.

Energy consumption during runs of default and optimized ML models, and during the optimization processes are compared. For all ML algorithms, optimized models have shown lower emission and energy consumption in comparison with the initial default models. In all cases, emissions during the optimization processes were higher by few orders of magnitude The optimization process sets the best parameters to maximize the accuracy of a ML model, but it is often time- and energy-intensive.

RESULTS AND DISSCUSSION

The database for supervised training was taken from the UC Irvine Machine Learning Repository site (archive) that represents a collection of continuous and ordinal data types. Linear Regression, k-Nearest Neighbors Regressor and Decision Tree Regressor are trained and optimized using this dataset. Data are used offline i.e. data are downloaded and stored locally, to avoid repetition and additional energy costs (from data transmission and data center compute loads).

To estimate the carbon footprint of machine learning runs, experiments were conducted using open-source Python software within the PyCharm environment, with the CodeCarbon library activated to monitor emissions. For our purpose CodeCarbon library is confirmed as the straightforward solution. Application of CodeCarbon tool was effortlessly, and the data are recorded in a separate file. CodeCarbon measurement starts simultaneously with the ML run and stops when ML run terminates. All ML runs were performed at hardware characteristics described in Table 1. Also, GoogleColab environment is used to compare CO_2 emissions.

Table 1. *Hardware characteristics*

CPU	Intel(R) Core(TM) i5-9500 CPU @ 3.00GHz
RAM	16GB
GPU	iGPU
Location	Serbia, SRB
OS	Windows-10-10.0.19045-SP0
Python_version	3.13.2
CodeCarbon version	2.8.2003
tracking_mode	machine

The hardware components that consume the most energy include: Central Processing Unit (CPU), Graphics Processing Unit (GPU), and RAM. Usually, a CPU uses more energy than RAM (Henderson, 2020). Contribution to GHG emission by other hardware elements, such as storage and network can be considered negligible. CodeCarbon tool cannot measure energy consumption of non-NIVIDIA GPUs. Estimation of energy consumption by hardware defined in Table 1, should be considered as a conservative (we assume that iGPU contribution is small). Results of energy consumed by hardware parts (CPU and RAM) during the Linear Regression runs (Figure 2) confirm the previous conclusion. The difference between CPU and RAM energy consumption is around one order of magnitude.

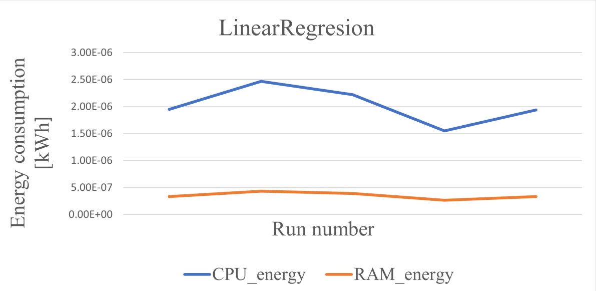


Figure 2. *Energy consumed by components during the Linear Regression run (values shown on the y axis in kWh). CPU energy consumption is around 10 times more than the energy consumed by RAM.*

Energy consumption during the run of default and optimized model, as well as the optimization process for the three applied ML algorithms are compared. To reach defined accuracy ML algorithm should be tailored to the problem that needs to be solved by tuning parameters.

Optimization process implies parameters adjustment to improve predictive performances, balance the bias-variance trade off, enhance generalization, and efficiency. Optimized models ensure stability and maintain efficiency in real-world applications. Generally, it performs faster and shows more sustainable features. However, the optimization process can be time and energy demanding. Is it worth to rise energy consumption to optimize ML models? It depends on the purpose. For intensive and long-term ML usage, optimization can be a more sustainable solution, because an optimized model in the long run can provide a smaller emission. If machine learning algorithms are intended for short-term use—such as in educational contexts like lectures or exercises—where high precision and extensive parameter tuning are not essential, it is worth considering whether model optimization is necessary or if the default configuration is sufficient to solve the given task.. ML algorithms with moderate complexity and small training set, and consequently negligible emissions, are very suitable for educational purposes, but there is no need to perform many iterations to achieve high precision.

Linear Regression is a simple ML model, easy to train. Energy consumption of Linear Regression training and testing is minor. In comparison of energy consumption by default and, optimized model, as well as the optimization process, it is obvious that the difference is not worthy of consideration. Negligible CO_2 emissions of Linear Regression make it very appropriate for multiple applications in different fields, regarding responsible and sustainable AI. It is appropriate for applications in education, industry, engineering, statistics, biostatistics, etc. In education, it is a core to statistics and introductory ML courses, and for teaching interpretability and inference models. Linear Regression has various applications in industry and engineering in forecasting, strategies, trend analysis, predicting system performance, control system modelling, thermal system modelling, stress-strain modelling in materials, etc. (Freedman, 2009; James, 2013). Also, it is applicable in high-dimensional biological and medical data.

Three parameters of the k-Nearest Neighbors Regressor are adjusted during the optimization process: n_neighbors, which determines model complexity; leaf_size, which controls the size of leaf nodes in tree-based neighbor searches; and p, which determines the distance metric used. The best performing model had values n_neighbors = 10 (the number of nearest neighbors), leaf_size = 20 (optimizes neighbor search efficiency without affecting accuracy), and p = 3 (Minkowski distance). Parameters n_jobs = 1 indicates runs on a single core (serial processing). The results obtained for both the default and optimized models indicate exceptionally low emissions (Figure 3). The accuracy of models with different parameters (default or the best) varies. The optimized model (with the best parameters) provides significantly higher accuracy. As we expected (Figure 3), the emission of optimization (orange line) is higher due to retraining. The optimized model (grey line) exhibits the lowest emissions, with the least execution time and energy consumption. These results confirm that processes of parameter tuning can be more carbon intensive, but the optimized model has one order of magnitude lower emissions than the default model. This conclusion is significant for the training of deep learning and large language models, as their emissions can be higher (Lacoste, 2019; Henderson, 2020; Strubell, 2019).

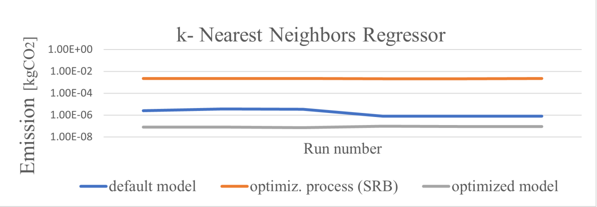


Figure 3. *k-Nearest Neighbors Regressor emissions of CO_2 during the runs of the default model (blue line), and optimized model (grey line), as well as during the optimization process (orange line).*

Bearing in mind that the k-Nearest Neighbors Regressor, produces negligible emissions, optimization for achieving high accuracy may be performed, without a meaningful impact on the environment. K-Nearest Neighbors is a straightforward algorithm commonly used in education, computational chemistry (for predicting molecular properties), medical imaging, remote sensing, nonlinear regression, and other fields. In education, it is suitable for teaching the concept of ML learning based on similarity, and for demonstrating overfitting and model complexity. The negligible carbon footprint of this algorithm encourages its applications in both industry and engineering for recommender systems, fraud detection, sensor data interpolation, and image processing.

Decision Tree Regressor follows the same pattern of emissions between default and optimized models and the optimization process (Figure 4). Parameters are adjusted to define a regularized, shallow decision tree that prioritizes robustness and generalization. The best suited parameters are criterion = absolute_error (the splitting criterion for evaluating the quality of a split, encourages robust predictions), max_depth = 2 (limits the maximum depth to 2 levels and reducing model complexity), and max_leaf_nodes = 3 (limits the number of terminal nodes and controls overfitting). This combination of parameters resulted in a straightforward and interpretable model that emphasizes robustness. The model with optimal parameters is significantly more accurate than the default model. The emission of the default model (blue line) is slightly higher than for the optimized model (grey line). As anticipated, the optimization process involves increased computational effort and results in higher emissions (represented by the orange line). Decision Tree Regressor optimization process shows a slightly higher emissions than k-Nearest Neighbors Regressor, but the accuracy of the optimized model is remarkably better than for the default. This algorithm requires adaptation to the problem by adjusting the parameters. Even though model optimization results in higher emissions, it should be optimized if the model is intended for long-term use. As previous ML models with negligible carbon emissions, this algorithm is also appropriate for education, especially for introducing concepts of model interpretability and as a base for learning more complex algorithms (for example, Random Forests). It is widely used across industries for purposes such as fault detection, performance prediction, medical diagnostics, ecology, and more.

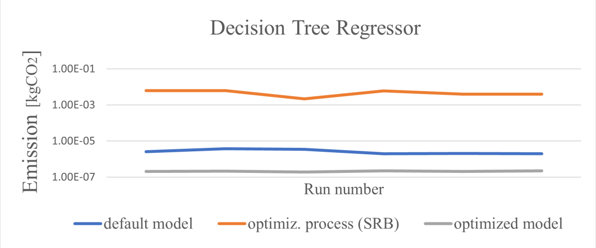


Figure 4. *Decision Three Regressor emissions of CO_2 during the runs of default model (blue line), and optimized model (grey line), as well as during the optimization process (orange line).*

Estimation of emissions and carbon footprint should draw attention to concerning current issue - the increasing use of AI in education, science, and in everyday life by millions of people. We usually approach computing resources as if they are limitless, but they are constrained in energy and environmental costs. This paper tracks the footprint of ML algorithms used to study basic principles of ML by solving specific tasks for a master's thesis. Emissions of default and optimized ML models are very small (< 1 µg CO_2eq), and the optimized models have one order of magnitude less emission than default, for both algorithms. Optimization processes for both algorithms (k-Nearest Neighbors and Decision Tree) perform more slowly, last longer, and show higher emissions (by several orders of magnitude). Higher emissions of optimization may have different origin that should be investigated (even if the model is lightweight, performing a large number of operations can extend runtime and memory usage, leading to higher overall energy consumption and emissions). If the optimization is run on a machine powered by a high-carbon-intensity grid (e.g., coal-heavy), even small increases in energy usage result in large increases in CO_2 equivalent emissions.

To compare emissions for different regions, we selected a country with a low carbon intensity value, Norway (Figure 5 and Figure 6). Figures 5 and 6 illustrate a decline in emissions by an order of magnitude when the location is changed from Serbia to Norway. The carbon intensity of electricity generation in Norway (0,031 kgCO2eq/KWh) is significantly lower than in Serbia (0,671kgCO2eq/KWh), primarily due to the predominance of renewable energy sources. Used ML algorithms are simple, and their optimization may not have a major impact on emissions, but when we deal with complex hyperparametric ML architectures at high carbon intensity locations, optimization becomes a concerning issue.

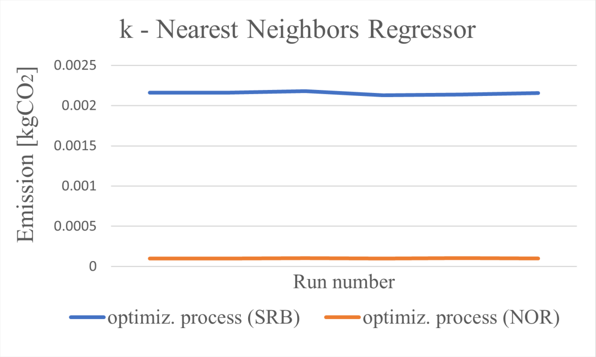


Figure 5. *Comparison of k – Nearest Neighbors Regressor emissions during the optimization processes at two selected locations: Serbia (blue line) and Norwegian (orange line).*

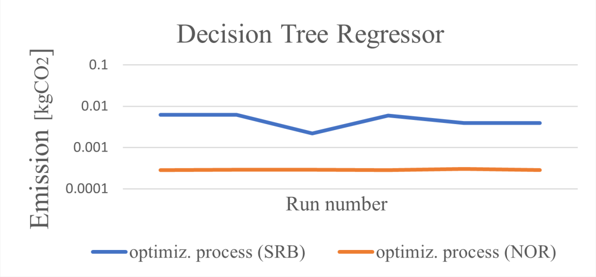


Figure 6. *Comparison of Decision Tree Regressor emissions during the optimization processes at two selected locations: Serbia (blue line) and Norwegian (orange line).*

Although energy consumption and emissions during the training and run of three applied ML algorithms are negligible, tracking carbon footprint is the first step toward fostering environmentally responsible AI. It should be kept in mind that the rapid development of computer technology and various algorithms can have notable environmental costs. In problem solving, scientists, practitioners, and other users typically emphasize accuracy without considering the energy expended during multiple iterations and trial-and-error processes. Each new run can be computationally intensive and energy demanding, particularly for complex algorithms with hyperparameters. Optimized models can be more accurate, and less energy intensive, but higher emissions during the optimization process are a critical issue. Complex algorithms may overfit, and optimization does not always guarantee improved accuracy. Numerous open questions regarding to parameter tuning, algorithm efficiency, and introduction of new ML practices require detailed analysis. Prioritizing cost-benefit or accuracy-efficiency issues is the first step in the responsible AI application. The AI life cycle's carbon footprint encompasses manufacturing, transport, distribution, dataset creation, transfer, and storage. Training AI models is just a small part.

To illustrate this statement GoogleColab is used as a cloud-based environment. GoogleColab is a widely used resource, with data centers located around the world. Cloud-based environments offer accessible platforms for ML experiments by supplying virtual machines in large-scale data centers. However, it is not possible to choose a server for data processing during the environment creation. In the GoogleColab environment, there are certain restrictions for the user (administrator rights, access to RAPL files, actual insight into hardware resources).

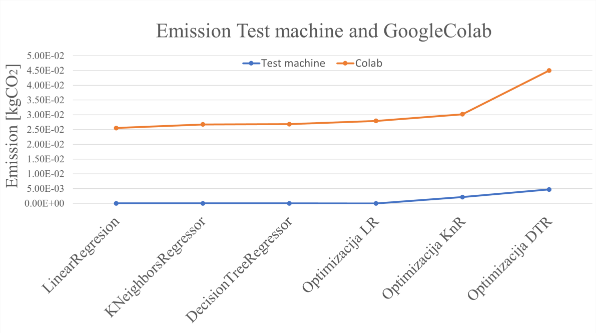


Figure 7. *Comparison of emissions for the test machine and GoogleColab during the runs of three ML algorithms and their optimizations.*

Emission report by GoogleColab includes energy consumption by full infrastructure: cooling, networking, data storage, etc. CodeCarbon estimates emissions based on actual energy consumption during the local execution of code, measuring CPU, GPU, and memory usage, along with execution time and regional carbon intensity of electricity (Lacoste et al., 2019). GoogleColab reports emissions for the entire duration that a virtual machine is distributed, regardless of the duration of actual computation runs. In our experiment, this virtual machine remains active for more than 40 minutes, even if the computation takes only seconds on the test machine. Decision Tree Regressor optimization took three times longer, and accordingly energy cost and emission had higher values. However, as shown in Figure 7, carbon emissions reported by GoogleColab are 2 orders of magnitude higher than those estimated by local tools CodeCarbon, even for short computations. This discrepancy is primarily due to different accounting, because GoogleColab estimates the true environmental cost of cloud resources, including both active and idle power usage. This estimation represents the hidden carbon costs of AI development and infrastructure (Strubell et al., 2019; Schwartz et al., 2020). A new sustainability challenge in ML practice is including emissions from passive infrastructure used in local measurements. It is of great importance, since data centers consume more electricity, accounting for 3% of global electricity supply and 2% of total GHG emissions, and this percentage is continuously growing (to 14% of the world's emissions by 2040, according to estimation of Thangam et al., 2024 and IEA).

Numerous questions about AI use need to be answered in the future, including carbon footprint over the life cycle, encompassing manufacturing, transport, product use and recycling (Wu, 2021; Guo, 2022).

CONCLUSION

In this paper, we estimated CO_2 emissions associated with the training of three commonly used machine learning (ML) algorithms Linear Regression, k-Nearest Neighbors Regressor, and Decision Tree Regressor, applied to educational tasks for a small dataset. We proposed CodeCarbon as a simple, easy-to-implement tool suitable for estimating the environmental cost of ML run. Our analysis showed that default models produce extremely low emissions (around 10^{-7} to 10^{-6} $kgCO_2eq$), and optimized models even lower (around 10^{-8} $kgCO_2eq$). However, the optimization process itself has higher environmental cost, with emissions rising by a few orders of magnitude (around 10^{-4} to 10^{-3} $kgCO_2eq$). Further, we compared CodeCarbon estimation with those from GoogleColab, a popular cloud-based platform. Emissions reported by Colab are consistently two orders of magnitude higher. This discrepancy can be explained by the resource distribution, idle compute supply, and infrastructure inherent to cloud services, which local estimators like CodeCarbon do not include. These results underline the importance of transparent, and conscious emission tracking. Responsible use of ML implies applications of low-impact computation (employing lightweight ML as it is satisfactory), local resources when it is feasible, minimizing unnecessary optimization cycles, and selection of algorithms that balance accuracy and energy efficiency. As AI becomes increasingly personalized and embedded into global infrastructure, responsible development must prioritize the performance/efficiency trade-off. Although the numerous algorithms (such as those used in this paper) have a minor carbon footprint, their scaling and increasing presence across daily life, science, industry, agriculture, medicine, business, etc., amplify overall energy requirements. Therefore, it is crucial to treat ML emissions not as a separate issue but within the broader context of the expanding digital environment.

Recommendations and Future Directions

The forward path requires intensive action toward the development of green, environmentally responsible AI, to ensure long-term societal and ecological feasibility. AI practitioners should pay attention to using available ML tools thoughtfully, avoiding unnecessary optimization, improving optimization efficiency, considering energy-efficient algorithms and hardware. In addition, they should select local computation over cloud platforms and use renewable energy sources and low-carbon regions when feasible. It would be particularly useful to analyze emissions during the ML run using tools such as CodeCarbon and report results in a research paper along with other results. Future work should be expanded and based on the analysis of different hardware environments, larger datasets, and a broader range of algorithms, including deep learning models.

ACKNOWLEDGEMENTS

This paper is supported by the Ministry of Science, Technological Development and Innovation of the Republic of Serbia pursuant to agreement № 451-03-137/2025-03/200148, goal 13.

REFERENCES

ALLEA (2022) *Towards Climate Sustainability of the Academic System in Europe and Beyond*. Berlin. ISBN: 978-3-9823967-1-2. DOI 10.26356/climate-sust-acad. https://allea.org/wp-content/uploads/2022/05/ALLEA-Report-Towards-Climate-Sustainability-of-the-Academic-System.pdf (accessed: May 2025)

Altman, N. S. (1992). *An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression*. The American Statistician, 46, pp. 175–185. https://doi.org/10.1080/00031305.1992.10475879

Alpaydın, E. (2010). Introduction to machine learning (2nd ed.). MIT Press. ISBN 978-0-262-01243-0

Alzoubi, Y. (2024). *Green artificial intelligence initiatives: Potentials and challenges*. Journal of Cleaner Production. 468, 143090. https://doi.org/10.1016/j.jclepro.2024.143090

Bannour, N., et al. (2021). *Evaluating the carbon footprint of NLP methods: a survey and analysis of existing tools*. In Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing, pages 11–21. https://aclanthology.org/2021.sustainlp-1.2/

Bergstra, J., et al. (2011). *Algorithms for Hyper-Parameter Optimization*. Advances in Neural Information Processing Systems. 24. Eds J. Shawe-Taylor, R. Zemel and P. Bartlett, F. Pereira, K. Q. Weinberger https://proceedings.neurips.cc/paper_files/paper/2011/file/86e8f7ab32cfd12577bc2619bc635690-Paper.pdf

Bouza, G., et al. (2023). *How to estimate carbon footprint when training deep learning models? A guide and review*. Environmental Research Communications. 5, 115014. https://iopscience.iop.org/article/10.1088/2515-7620/acf81b

CodeCarbon. https://codecarbon.io/ (accessed: March 2025)

https://mlco2.github.io/codecarbon/index.html (accessed: March 2025)

Dayarathna, D., et al. (2015). *Data center energy consumption modeling:A survey*. IEEE Communications Surveys & Tutorials, 18, pp. 732–794. DOI: 10.1109/COMST.2015.2481183.

Dodge, J., et al. (2022). *Measuring the Carbon Intensity of AI in Cloud Instances*. Proceedings FAccT '22: Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency June 21 - 24. https://doi.org/10.1145/3531146.3533234.

Electricity Maps. https://www.electricitymaps.com/

Ember (2025); Energy Institute - Statistical Review of World Energy (2024). https://ember-energy.org/latest-insights/global-electricity-review-2025/(accessed: May 2025)

Freedman, D. (2009). *Statistical Models: Theory and Practice* 2nd. Cambridge University Press. https://doi.org/10.1017/CBO9780511815867.

Google Colab, https://colab.research.google.com/(accessed: March 2025)

Guo, B., et al. (2022). *Energy-efficient database systems: A systematic survey*. ACM Computing Surveys, 55, 1–53. https://dl.acm.org/doi/10.1145/3538225

Henderson, P., et al. (2020). *Towards the Systematic Reporting of the Energy and Carbon Footprints of Machine Learning*. Journal of Machine Learning Research 21 pp. 1-43. https://jmlr.org/papers/v21/20-312.html.

The International Energy Agency (IEA) https://www.iea.org/energy-system/buildings/data-centres-and-data-transmission-networks?mod=livecoverage_web&utm_source=chatgpt.com (accessed: May 2025)

IPCC. (2023). Climate Change 2023: Synthesis Report. Contribution of Working Groups I, II and III to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change [Core Writing Team, H. Lee and J. Romero (eds.)]. IPCC, Geneva, Switzerland, 184 pp., https://www.ipcc.ch/report/ar6/syr/downloads/report/IPCC_AR6_SYR_FullVolume.pdf

James, G. et al. (2013). *An introduction to statistical learning* 112. New York: Springer. ISBN 978-1-0716-1417-4 https://doi.org/10.1007/978-1-0716-1418-1

Kaack, L. H., et al. (2022). *Aligning artificial intelligence with climate change mitigation*. Nature Climate Change 12, pp. 518–527. https://doi.org/10.1038/s41558-022-01377-7

Lacoste A, et al. (2019). *Quantifying the carbon emissions of machine learning*. https://arxiv.org/abs/1910.09700

Lannelongue, L., et al. (2021). *Green Algorithms: Quantifying the Carbon Footprint of Computation*. Advanced Science. 8, 2100707. https://doi.org/10.1002/advs.202100707

Lannelongue, L., & Inouye, M. (2023). *The carbon footprint of computational research*. Nature Computational Science 3, 659. https://doi.org/10.1038/s43588-023-00506-2

LEAFframeworkTake part in LEAF | Sustainable UCL - UCL –University College London https://www.ucl.ac.uk/sustainable/case-studies/2020/aug/take-part-leaf (accessed: May 2025)

Luccioni, A., et al. (2023). *Estimating the carbon footprint of BLOOM, a 176B parameter language model*. The Journal of Machine Learning Research. 24 pp. 11990 - 12004 https://dl.acm.org/doi/10.5555/3648699.3648952.

Olawade, D., et al. (2024)a. Artificial intelligence in environmental monitoring: Advancements, challenges, and future directions. Hygiene and Environmental Health Advances, 12, Article 100114. https://doi.org/10.1016/j.heha.2024.100114

Olawade, D., et al. (2024)b. Artificial intelligence potential for net zero sustainability: Current evidence and prospects. Next Sustainability, 4, 100041. https://doi.org/10.1016/j.nxsust.2024.100041.

Patterson, D. et al., (2021). *Carbon Emissions and Large Neural Network Training*. Machine Learning. https://doi.org/10.48550/arXiv.2104.10350.

Rolnick, D., et al., (2019). Tackling climate change with machine learning. ACM Computing Surveys. 55 pp. 1-96. https://doi.org/10.1145/3485128

Schwartz, R. et al., (2020). *Green AI*. Communications of the ACM, 63, pp. 54-63. https://doi.org/10.1145/3381831.

Strubell, E., et al. (2019). *Energy and Policy Considerations for Deep Learning in NLP*. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, p. 3645–3650, Florence, Italy. https://aclanthology.org/P19-1355/.

Strubell, E., et al. (2020). *Energy and policy considerations for modern deep learning research*. Proceedings of the AAAI Conference on Artificial Intelligence, 34, pp. 13693-13696. https://doi.org/10.1609/aaai.v34i09.7123.

Thangam, D., et al. (2024). *Impact of Data Centers on Power Consumption, Climate Change, and Sustainability*. In K. Kumar, V. Varadarajan, N. Nasser, & R. Poluru (Eds.), Computational Intelligence for Green Cloud Computing and Digital Waste Management (pp. 60-83). IGI Global Scientific Publishing. https://doi.org/10.4018/979-8-3693-1552-1.ch004

UC Irvine Machine Learning Repository site https://archive.ics.uci.edu/dataset/320/student+performance

(accessed: March 2025)

Vinuesa, R., et al. (2020). The role of artificial intelligence in achieving the Sustainable Development Goals. *Nature Communications*, 11, p. 233. https://doi.org/10.1038/s41467-019-14108-y

Wang, J., et al. (2022). Sustainable AI: Environmental implications, challenges and opportunities. Machine Learning and Systems, 795–813. https://arxiv.org/abs/2111.00364.

Xing, E. & Monck, A. (2023). *AI emissions are fueling a new doomerism this time it's climate change.*

https://fortune.com/2023/12/12/ai-emissions-doomerism-climate-change-environment-tech/

Wu, C.-J., et al. (2022). *Sustainable AI: Environmental implications, challenges and opportunities*. Machine Learning and Systems. https://arxiv.org/abs/2111.00364

XSEDE Impact – Usage Statistics. (n.d.). Retrieved May 2025, from https://portal.xsede.org/#/gallery

(accessed: May 2025)