Division of Computing Science and Mathematics
Faculty of Natural Sciences
University of Stirling

Towards Faster, Greener AI

Kwame Ntim Duodu

Dissertation submitted in partial fulfilment for the degree of
Master of Science in Artificial Intelligence

September 2021

Abstract

The application of Machine Learning (ML) and Artificial Intelligence (AI) in a broader sense has seen tremendous acceptance and increase in recent times. The development of autonomous vehicles, contact tracing apps, diagnosis of diseases based on symptoms, detection of fraudulent transactions among a host of other applications are just a few of the remarkable feats of AI/ML. However, much emphasis on various AI/ML projects is placed on building models with higher predictive accuracies, ignoring the associated environmental cost. As AI/ML thrives on data stored in data centres with ever-increasing electricity usage, coupled with the battery-power constraint of the smart devices used by end-users which require frequent charging, the huge quantities of electricity usage leading to the emission of Greenhouse Gases (GHG) with its subsequent effects of climate change and global warming cannot be overlooked. With the fight towards net-zero emissions prominently featuring in global policies, the need to build AI/ML models which are eco-friendly while seeking to maintain predictive accuracy is therefore non-negotiable.

This dissertation explored hyperparameter optimisation and feature optimisation as techniques to employ in building faster and greener AI models while keeping predictive accuracy at bay. GridSearchCV and RFECV as provided by scikit-learn were used for the hyperparameter optimisation and feature optimisation respectively in this research by applying the eXtreme Gradient Boosting (XGB) classification algorithm on the Smart Grid Stability (SGS) dataset to assist in achieving the set objectives. It was found that it required 393times the inference runtime and 701times the inference energy for a model to be trained. Following this finding, it was also revealed that the runtime and energy consumption at the training phase of the ML cycle could be cut down by 30.88% and 35.07% respectively with a minimal loss of 1.93% in predictive accuracy from 97.8% to 95.91% when hyperparameter optimisation is applied on selected features of relevance. The findings of the research, therefore, confirm the need to employ hyperparameter optimisation and feature optimisation targeting energy as well as accuracy, as valuable tools towards faster and greener AI.

Attestation

I understand the nature of plagiarism, and I am aware of the University's policy on this.

I certify that this dissertation reports original work by me during my university project except for the following):

- The pyRAPL technology for the energy consumption used in this dissertation and the fundamental ideas of this dissertation was largely adopted from [1].

Signature: 

Date: 24th September, 2021

Acknowledgements

I am very much grateful to my supervisor, Dr. Alexander Brownlee, whose expertise was invaluable in completing this research work.

My heartfelt gratitude to my wife Esther and my four lovely girls, Pokuaa Panin, Pokuaa Kakra, Akyaamaa Panin and Akyaamaa Kakra for the love and encouragement towards the completion of this research.

Finally, to my mum, family and friends who were of immense support in various capacities just to ensure this work is well executed.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AGI	Artificial General Intelligence
AI	Artificial Intelligence
ANI	Artificial narrow intelligence
ASI	Artificial superintelligence (ASI)
CO2	Carbon Dioxide
CPU	Central Processing Unit
CV	Cross Validation
DRAM	Dynamic Random Access Memory
FS	Feature Selection
GHG	Greenhouse Gas
GPS	Global Positioning System
GPU	Graphics Processing Unit
IPCC	Intergovernmental Panel on Climate Change
HAR	Human Activity Recognition
ML	Machine Learning
MLP	Multilayer Perceptron
NLP	Natural Language Processing
NN	Neural Network
RAPL	Running Average Power Limit
RFECV	Recursive Feature Elimination with cross Validation
SBSE	Search-based Software Engineering
SGS	Smart Grid Stability
UK	United Kingdom
UN	United Nations
WEKA	Waikato Environment for Knowledge Analysis
XGB	eXtreme Gradient Boosting

1 Introduction

This first section of the dissertation sets out the background and context based on which this research was carried out. Following the background and context, the scope and objectives of the research are defined. A summary of achievements stemming from the research comes next and finally, a general overview of how the whole dissertation is organised is made known to assist with easy navigation of this dissertation.

1.1 Background and Context

Modern Artificial Intelligence (AI) and the narrow field of Machine Learning (ML) has been with us for over 60 years now but in recent times, its application in industry and acceptance by the masses has increased tremendously [2]. In present-days, many devices have been developed and others are still being developed to exploit machine learning in the healthcare industry, agricultural industry, financial industry, manufacturing industry, logistics industry and several other industries [3]. For instance, in the wake of the Covid-19 pandemic, AI and ML were considered one of the key tools to be used in battling the disease; devices that are capable of detecting who has the disease based on symptoms and other contact tracing applications rely on AI and ML algorithms [4]. As intelligent devices rely on data, which is one of the essential components in their design and ability to operate, it is of great importance that vast server centres are developed to house these large-scale datasets. The ever-increasing quantities of energy consumed by these large-scale data centres, coupled with the need to frequently charge the designed intelligent devices used in accessing resources from the data centres due to the energy taken up by their Central Processing Units (CPUs) in carrying out their intelligent tasks, are known to contribute significantly to the soaring rates in Carbon Dioxide (CO_2) emissions leading to the greenhouse effect which is detrimental to our planet [5], [6].

Expanded concentration of CO_2 in the atmosphere is generally being considered as the primary driving component that causes global warming and the climate change phenomena [7]. Although, global warming and climate change are inevitable, environmental pollutions originating in human activities contribute approximately 1.0 degree Celsius to this global canker as reported in 2018 by the Intergovernmental Panel on Climate Change (IPCC) [8]. Some of the numerous consequences of this global warming and climate change menace include; higher temperatures leading to many disasters such as storms, hurricanes, cyclones, Arctic ice coverage reduction and prolonged droughts leading to fire outbreaks such as the one evidenced in Australia recently. The fight towards *net-zero* emissions continues to feature prominently among government policies the world over. The United Kingdom (UK) government in 2019 for instance, amended its Climate Change Act of 2008 [9] with a legislation [10] to fuse a net-zero emissions goal for 2050, as opposed to the 80% reduction target benchmarked to the 1990 emissions previously set.

Undoubtedly, electricity demand for computations in data centres and the use of electricity to frequently get batteries of devices relying on AI technologies charged, due to the higher energies taken up by CPUs of such devices, cannot be ignored. This ever-increasing electricity demand all around the world and its accompanying upsurge in CO_2 emissions has an environmental impact that is becoming increasingly harder to overlook. As data centres are anticipated to devour 8% to 21% of worldwide power by the year 2025, it is predicted that training of a large-scale deep-learning model would emit 284,000kgs of the green-house CO_2, identical to the combined lifetime emissions of 5 automobiles [11]. On the other part, the smart devices which in most cases are mobile in nature and used in interacting with the data centres are built with batteries which have the capacity to hold more power. Despite these smart devices' battery power-holding capacities, while features such as screen display, GPS, Wi-Fi consume large portions of power, the standalone battery power consumption of these devices CPUs utilisation eats up between 27% and 35% of a fully charged device's battery [12].

Currently, the move to build AI and for that matter ML systems with faster compute and which are less costly to planet earth, have aroused the interests of many researchers in the AI community. Researchers have tried to explore techniques capable of making AI systems perform intelligent tasks accurately and faster in order to reduce the overall energy consumption and its alarming effects on the environment. In light of this, optimisation techniques readily come to mind. However, as to which aspect of the machine learning process to modify in the quest of building faster, accurate and energy efficient AI systems at both extremes of training and inference are being explored by researchers.

The application of optimisation techniques to configure hyperparameters has been used extensively in almost every AI/ML project to build models which are more accurate and, in some cases, reduced the computational time. It is worrying to note however, that only a handful of researchers have investigated the extent to which hyperparameter optimisation can be used as an approach in constructing models that are computationally fast and energy efficient by measuring the runtime and energy consumptions to ascertain the energy efficiencies of the application of hyperparameter optimisation. Some of these works include [1], [13]–[16] . Another aspect of the machine learning process, where the use of optimisation techniques has been indispensable is Feature selection. Feature selection besides its ability to increase model performance is widely appraised also to be helpful in building simpler and faster machine learning models. This it achieves by using a subset of the dataset's features instead of the entire features [17]. However, preliminary research of literature indicated that much study has not been done by measuring the runtime and energy consumption to establish the true impact of feature optimisation on runtime and energy consumption with its subsequent effect in reducing the GHG. One key research that employed feature optimisation as a tool to reduce energy consumption in wearable devices is the work of Ghasemzadeh et al. [18].

In the most related preceding work by Brownlee et al.[1] which forms the basis of this dissertation, the researchers explored the energy consumption using search-based approach for hyperparameter tuning of MLP on five classification datasets. The research motive was clearly stated as they focused on the investigation of the trade-offs between training or inference energy and classification accuracy. The researchers were able to demonstrate based on one of their datasets that 77% of inference energy consumption could be saved with a minimal reduction in prediction accuracy from 94.3% to 93.2%. The authors further noted that, training energy consumption could be lessened by 30% to 50% with a slight loss in model accuracy. However, the researchers focused on hyperparameter optimisation technique only. Again, their research was based on multilayer perceptron (MLP) machine learning algorithm, five specific datasets and with the experimentation being performed on a specific hardware system to make generalisations [1].

To further confirm the generalisations of the researchers and add on to the existing knowledge of literature, it is essential that research be carried out utilising a different dataset and different ML algorithm on a different hardware architecture. Extending the work of Brownlee et al. [1] further, investigating the runtime and energy consumptions stemming from a combination of hyperparameter optimisation and feature optimisation techniques in the development of environmentally friendly models while keeping accuracy at bay, cannot be overemphasised. In the light of this, the following research questions are formulated:

1. What is the runtime and energy consumption associated with building an accurate machine learning model, using the default hyperparameter configuration of the ML algorithm on the entire features of the dataset under consideration, at both extremes of training and inference?

2. To what extent, can hyperparameter optimisation be used to reduce runtime and energy consumption of ML algorithm while preserving accuracy at both extremes of training and inference?
3. Is feature optimisation useful in trying to build machine learning model that is energy efficient and computes faster during training and inference with little or no disruption in accuracy?
4. How fast, and how much energy can be saved when hyperparameter optimisation is applied on selected features of a dataset in building machine learning models while preserving accuracy during training and also at inference?

1.2 Scope and Objectives

Whilst hyperparameter optimisation and feature selection techniques have been used extensively in machine learning projects, research on their contribution to building eco-friendly AI systems is still green. The primary focus of this work is set to investigate hyperparameter optimisation and feature optimisation as techniques capable of making machine learning algorithms environmentally friendly by cutting down on runtime and energy consumption at both ends of training and inference without necessarily causing harm to the predictive accuracy of the model being constructed.

The scope of this research is limited to the supervised machine learning task of classification. Focus will be on the eXtreme Gradient Boosting (XGB) classifier, due to its performance and popularity on classification problems. More so, focus will be on the Smart Grid Stability (SGS) dataset [19]. These limitations in part, are as a result of the limited timeframe and computational resource within which this research is to be carried out considering the computational cost (both environmentally and economically) associated with AI projects. Moreover, the energy implications of this research are geographically limited in scope to the UK as other countries may have a mix of electricity generating sources other than that of the UK. Quantification of the environmental cost may be different if this research is carried out in a different geographical location.

In an attempt to answer the research questions formulated and also to achieve the overall aim of making AI faster and greener, the following specific objectives were set:

1. To investigate runtime and energy consumption of XGB classification algorithm on the SGS dataset at both extremes of training and inference.
2. To explore the impact of hyperparameter optimisation on the runtime and energy consumption of XGB classification algorithm on the SGS dataset during training and inference.
3. To examine the effects of feature optimisation on runtime and energy consumption of XGB classification algorithm on the SGS dataset at both ends of training and inference.
4. To explore the combined effect of hyperparameter optimisation and feature optimisation on the runtime and energy consumption of XGB classification algorithm on the SGS dataset at both extremes of training and inference.

1.3 Achievements

It is worth assessing to what extent the series of experimentations in this dissertation has helped in achieving the set objectives, answered the formulated research questions and also, contribute to existing knowledge on the broader aim of exploring techniques appropriate in the quest of making AI faster and environmentally friendly.

With regards to objective 1, runtime and energy consumption at both extremes of training and inference were reliably measured, serving as a benchmark for comparison and also revealing at

which extreme much energy is used. In the case of objective 2, hyperparameter optimisation drastically cut down the runtime and energy consumed on training the model, however, the cut observed for inference was seen to be very small but tested to be statistically significant. Unlike objective 2, the experimental results for objective 3 only showed a marginal drop in runtime and energy consumption, however, the drop was tested to be statistically significant. The final objective, objective 4 was achieved as savings in runtime and energy consumption for both training and inference were recorded but observed to be very substantial at the training extreme. It is worth noting that minimal losses in predictive accuracy for all the series of experimentations conducted were observed.

The findings confirm that although hyperparameter optimisation on its own is a great technique to apply in building faster and environmentally friendly AI systems, however, the combined effect of hyperparameter optimisation and feature optimisation is superb to aid in constructing ML models that are accurate, environmentally friendly and also computationally faster.

This dissertation has helped with the understanding and knowledge of the pyRAPL software toolkit. I am now able to confidently use it in projects to measure the energy footprint of a host machine together with the execution of a piece of python code.

1.4 Overview of Dissertation

The dissertation is organised into five main sections. Below is an overview of how this dissertation has been structed:

Section 1 – Introduction: This section of the dissertation presents the general background and the need for the research. The scope and objectives of the study are clearly stated in this section of the dissertation. The section concludes with a brief and clear summary of the achievements of this dissertation.

Subsequent sections of the dissertation are organised as below;

Section 2 – State-of-the-Art: The state-of-the-Art section of the dissertation is thematic in nature. It tries to bring to light, some of the current developments on key topics relating to this dissertation. Faster AI and the environmental cost, Optimisation techniques for faster and energy efficient AI, Hyperparameter optimisation for faster and energy efficient AI, Feature optimisation for faster and energy efficient AI and a blend of hyperparameter optimisation and feature optimisation for faster and energy efficient AI, are the thematic topics reviewed in this section. A critical literature review of the most related work by Brownlee et al. [1] concludes the section.

Section 3 – Methodology: The method and materials used for the experimentation is made known in this part of the dissertation in order to assist others who may like to replicate the processes involved, to seamlessly do so. The section is further expanded into the following subsections; Research design, Dataset, General approach, Measurements and Technology and systems used.

Section 4 – Results and Discussion: The results of the entire experimentations embodying this dissertation are presented in this section. Results of the individual experimentations are presented, followed by the general results with a thorough discussion of the findings of this dissertation.

Section 5, Conclusion: This section of the dissertation presents a snapshot and evaluation of the entire dissertation and also recommendations for further research on the topic.

2 State-of-The-Art

This section summarises state-of-the-art on the thematic topics of this dissertation. More so, it presents a critical literature review of the most related research preceding this dissertation. The section is expanded under the following thematic topics; Faster AI and the environmental cost, Optimisation techniques for faster and energy efficient AI, Hyperparameter optimisation for faster and energy efficient AI, Feature optimisation for faster and energy efficient AI and A blend of hyperparameter optimisation and feature optimisation for faster and energy efficient AI. The section concludes with a critical review of the most related research by Brownlee et al. [1] preceding this dissertation.

2.1 Faster AI and The Environmental Cost

In order to appreciate the concept of faster and greener AI, this subsection seeks to briefly delve into the underlying concepts of Artificial Intelligence/Machine Learning as well as Climate Change/Global warming. Past and present studies with focus on AI's contribution to global warming and its ensuing effects on our planet are also discussed.

2.1.1 Artificial Intelligence/Machine Learning

Numerous activities requiring the use of the human mind such as understanding language, driving a car, developing a computer game and the ability to apply common-sense in decision-making among others require the application of *"intelligence"*. Over the past few years, computer systems have been developed to act intelligently like the human mind, capable of performing these tasks. Self-driving vehicles, devices capable of diagnosing diseases, systems that are able to understand portions of natural language text and human speech, are just a few specific instances of systems built primarily to possess some degree of *artificial intelligence* [20].

Attempts to define AI robustly or in its simplest terms continues to be a challenge as there seem not to be a one fits all definition. One of the pioneers in AI, *John McCarthy* in 1995 made the first attempt in defining AI as:

"The goal of AI is to develop machines that behave as though they are intelligent"

However, the above definition seems not to be sufficient as proven by other researchers over the years. Several other researchers have tried to define AI but one that stands the test of time is the concise and terse definition by *Elaine Rich and Kevin Knight*. They defined AI as follows:

"Artificial Intelligence is the study of how to make computers do things at which, at the moment people are better."

Their definition of AI is apt and for years to come will still be relevant. What this definition reveals is that, the development of AI systems generally thrive on sound knowledge of human intelligence and reasoning [21].

Although, Artificial Intelligence as practiced today has existed over 6 decades, its application in industry and acceptance by the masses has increased tremendously in the recent past decade and now [1]. In present-days, many intelligent devices have been developed with others still being developed to assist in completing tasks in the healthcare industry, agricultural industry, financial industry, manufacturing industry, logistics industry and several other industries [2].

Artificial intelligence in the broader sense may be classed into three main types;

1. *Artificial narrow intelligence (ANI)* – AI systems with narrow range of abilities.
2. *Artificial general intelligence (AGI)* – AI systems at par with human capabilities.
3. *Artificial super intelligence (ASI)* – AI systems with capabilities beyond that of human.

It is generally known and accepted by the above definitions of the three broader types that, only ANI is currently being explored both in academia and industry. There are many subfields of AI under the ANI class but the following in no particular order constitute the six major subfields; Machine learning, Neural networks, Robotics, Expert systems, Fuzzy logic and Natural language processing.

Machine learning undoubtedly is one of the most popular and important branches of AI especially with its deep learning approach which started in the early parts of the immediate past decade. It addresses the issue of building computer systems that base on experience to automatically improve. Machine learning is one of the technical fields that has seen rapid growth in present times. It is at the core of AI and data science, interfacing statistics and computer science. The acceptance of data-intensive ML approaches is evident in technology, science and commerce, resulting in improved decision-making as observed across many spheres of life, including financial modelling, healthcare, policing, manufacturing, marketing, education and other sectors [22]. *Figure 1* below, gives a pictorial overview of the distinction between Artificial Intelligence and one of its branches, Machine Learning.

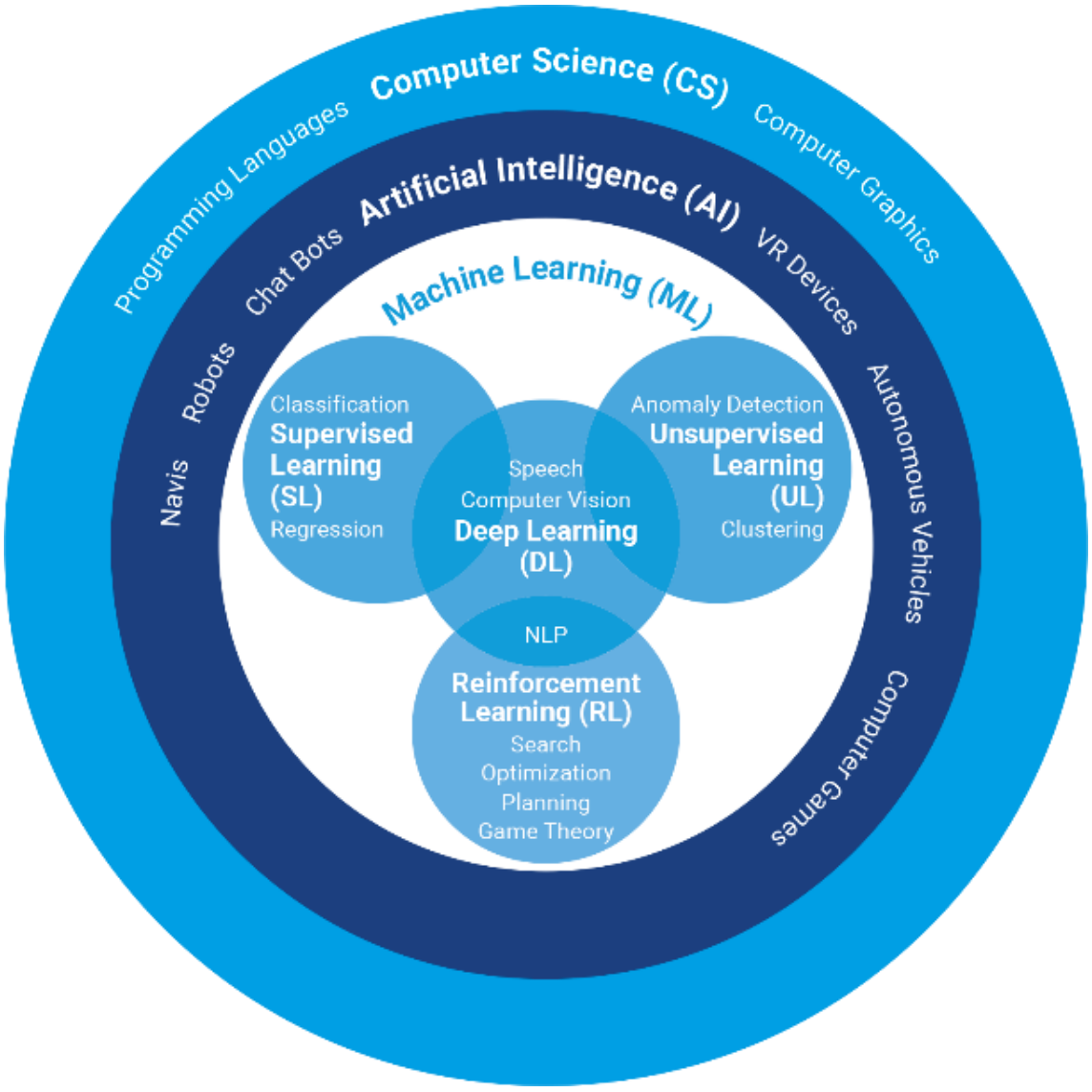

Figure 1. The Broader Picture of AI and Machine Learning

2.1.2 Climate Change and Global Warming

Climate change in its simplest terms as defined by National Geographic Society (NGS) is the persistent alteration in regional or global patterns [23] whereas Global warming has been phrased to mean, human activities impact on the climate. Some of the human activities leading to global warming encompass large scale deforestation and the excessive burning of fossil fuels which emit GHGs. Key among the GHGs emitted and worthy of note is CO_2. Climate change stems from global warming as the emission of GHGs into the atmosphere soak up emitted earth's surface infrared radiations, acting as a heavy surface covering to keep planet earth warmer than it should have naturally been. Most observable consequences of climate change on the environment include shrunken glaciers, premature flowering of trees, extinction of some plant and animal species, early break up of ice on lakes and rivers, and the list is unexhaustive. It is projected that, the effects of climate change will be with us into the foreseeable future; Artic becoming ice-free, rising sea levels of between 1ft to 8ft by 2100, Hurricanes becoming more intense and stronger, heat waves and droughts are cited as some of the expected consequences of climate change [24]. Global warming is a critical environmental issue whose consequences affect all and sundry and for this, there is a clarion call the world over to reduce the emissions of human-caused GHGs. One of such key calls worthy of note at the international levels is *"The Paris Agreement"* in 2015 by the UN. This international treaty which is legally binding on climate change was embraced in 2015 and enforced in 2016 by 196 parties. The main goal of the treaty is to restrict the global warming to below 2.0 degrees Celsius, preferably to 1.5 degrees Celsius relative to the levels that existed during the pre-industrial ages. It enforces partners to aim at reaching global peaking of GHGs emissions as soon as practicable so as to reach a climate-neutral world by 2050 and in effect contribute to achieving the overall goal [25]. The concept of global warming is illustrated in *Figure 2* below.

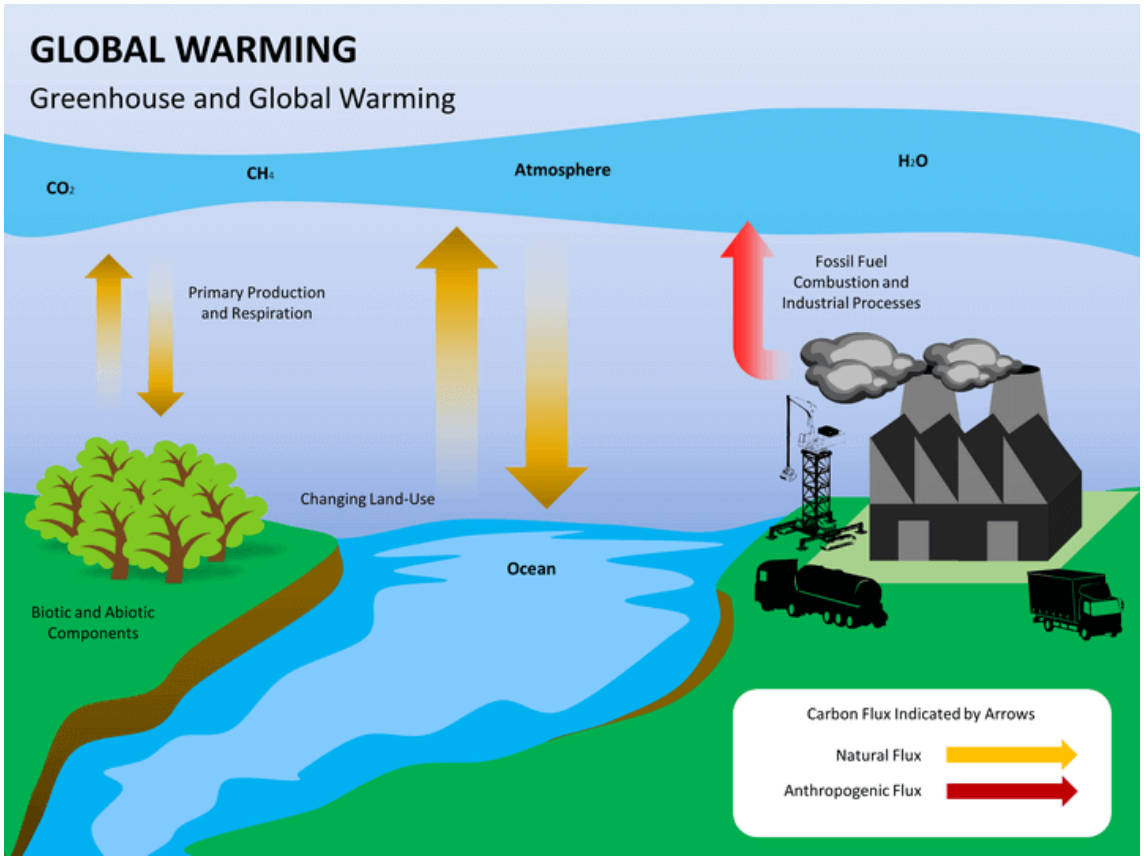

Figure 2. The Green-house effect and Global Warming

2.1.3 Faster AI and the Environmental Cost

The desire to build AI systems with near perfect predictive accuracies has been increasing rapidly in recent times especially with the introduction of deep-learning algorithms. Even though, awareness of energy consumption is gaining popularity in the AI community, almost every AI project still seem to be geared towards higher predictive accuracy as the prime performance metric reported in the end, ignoring the computational resources required, the environmental cost and with absolute disregard for sustainability [15]. It was estimated in 2010 that, out of the total electricity produced globally, data centres consume between 1.1% and 1.5% [26]. To give credence to the energy consumption levels at both extremes of data centres and smart devices used in accessing resources from these datacentres, global electricity consumption of between 8% to 21% by 2025 was predicted for datacentres. Again, an estimated 284,000kgs of CO_2, a GHG which is costly to our environment is emitted and this is only attributable to training of larger deep-learning algorithms. This emission compares to the combined lifespan emissions of 5 automobiles [11]. On the other extreme, which is in respect of smart devices, especially mobile phones used in accessing resources from datacentres, while user experience remains as a priority with regards to the factors that drain battery power [27], the CPU of these devices used in internal computations cannot be ignored as this feature prominently in higher batter usage [28]. For instance, in 2013 Carroll and Heiser [29] observed that, the highest CPU energy consumption of Samsung Galaxy S3 was 2,845mW. What this means is that, the peak energy consumption of the CPU exceeded that which was used by the screen and 3G hardware by 2.53times and 2.5times respectively.

Modern-day researchers in the AI community, while trying to achieve higher predictive accuracies are now aware of the associated computational and environmental costs. Although the level of awareness is questionable or at best research focus on the issue of building eco-friendly AI models could best be considered as very green as there seem to be limited literature on the topic. Some level of work has been done to bring the issue of AI related energy consumption and explored mechanisms for cutting down this worrying concern to the fore. Notable among such works include; [1], [5], [11], [13]–[16], [18], [26], [30]–[42]. In spite of the great works done by these researchers in attempting to explore how AI systems could be made to run faster and be made more energy efficient, it was noted during the literature review however, that the direct consequence on the environment as could have been measured by the level of actual GHGs emitted during their experimentations were ignored. This could be a problem attributable to a wanting of a standardised measuring technique or the different platforms on which these experimentations were set up. For example, whereas Brownlee et al. [13] estimated energy consumption on a Java-based platform using WEKA, Brownlee et al.'s [1] experimentation was done with python's RAPL provided by Intel. Further to this, whilst others were exploring energy consumption cut at both extremes of training and inference, other researchers too only focused on one side of the two larger but interlocking ends. It is worth noting that of the attempts made in exploring energy consumption measurements, most studies used the CPU time as a proxy for energy consumed for which other researchers have proven as an inappropriate measuring approach. Patterson et al. [37], in a most recent work formulated the energy consumption and its related carbon footprint of training a large Natural Language Processing (NLP) model. They argued that, the electricity demanded to allow a machine learning model to run is a function of the number of processors on the machine setup for the experimentation, the algorithm, the programming language that executes the task, the power and speed of the processors on the machine setup for the task and the datacentres energy sources mix (fossil, renewable energy, etc.) They then formulated simply, the effect of AI model on the environment as;

$$Footprint = (electrical\ energy_{train} + queries \times electrical\ energy_{inference}) \times CO_{2\ datacentre}/KWh$$

Albeit, this equation seems insightful, its validity is yet to be established by other researchers. Moreover, its application to other ML subfields other than the NLP is currently not known.

Optimisation techniques have been the main tool that has been employed on various aspects of AI and for that matter ML projects to construct optimal models that compute faster while preserving their accuracy in this era of power-aware AI. The ensuing subsection takes an empirical look at the application of optimisation techniques on the move towards faster and greener AI.

2.2 Optimisation Techniques for Faster and Energy Efficient AI

In every aspect of our daily lives, we are faced with the task of making the most effective or best use of resources or of a situation; this is what is referred to as optimisation. In present times, computing and mathematics have been connected in attempts to solve optimisation problems. Optimisation techniques have seen major applications in practice especially in commerce, industry, government and science to help in solving network problems, scare resources allocation problems, maximisation or minimisation of functions among others. Optimisation is at the heart of computer science. In software development for instance, optimisation techniques are employed so as to be able to build systems that are efficient, that is with faster runtime considering the limited computational resources available. It is significant to note that, optimisation is a fundamental block to artificial intelligence and for that matter machine learning. For example, optimisation techniques have made it possible to design artificial intelligence or machine learning algorithms that have been used to build self-repairing software, computers that interact with humans and also for data predictions and analysis among other applications [43], [44].

It is therefore of great essence that optimisation techniques are the go-to approaches in building artificial intelligence and machine learning algorithms which are computationally faster, energy efficient and in effect eco-friendly which achieves almost same or a minimal loss in accuracy as would have been achieved by a computationally expensive system. Several researchers in the Artificial Intelligence community have begun to explore search-based optimisation techniques to help in building systems that are energy efficient [1]. Notably, Brownlee et al. [1] applied grid-search optimisation technique to explore the hyperparameter settings for a MLP machine learning algorithm on 5 classification datasets with the main focus of minimising training or inference energy consumption at the expense of accuracy. More so, search-based software engineering (SBSE) optimisation technique was applied to reduce the energy consumed by programs written for Java virtual machines Brownlee et al. [13]. In all of these two instances where optimisation techniques were used, the set objectives of achieving energy efficient systems were achieved. In the case of [1], one of their experimentations resulted in an energy efficient system with a reduced 77% inference energy consumption whiles the reduction in accuracy was minimal, reduced to 93.2% from 94.3%. This same work also demonstrated the power of optimisation techniques' ability to cut down on energy consumption by a margin of between 30% and 50% when fitting a model to a training data with marginal loss in model accuracy as observed by the researchers. As also noted in the research work of [45], search-based optimisation technique was used to fix energy leakages in mobile devices and the researchers observed that the repair expressions generated by their system could save 60% of the energy consumption on tested apps.

Unfortunately, only few literatures exist on the application of search-based optimisation techniques for faster and energy efficient AI. These researchers [14], [16] also applied search-based optimisation techniques to reduce model complexity so as to be able to cut down the runtime and energy consumption. In the case of [14], the researchers proposed an energy efficient neural network structural design, Chameleon. According to their results, altering computational resources to building blocks is crucial to the performance of the model. Accuracy of 73.8% and 75.3% top-1 accuracy were achieved on ImageNet with reduced latency of 8.2% and 6.7%. On the other hand, [16] proposed HyperPower to tackle the issue of complex NN architectures as a solution to building energy efficient models by applying a Bayesian optimisation and random

search approaches. The researchers demonstrated that, the number of evaluation function and the best test error could be at a faster rate of 30.12times.

Machine learning is a process encompassing many stages such as data preparation, choosing the model, training the model, evaluating the model, hyperparameter tuning and finally making predictions. Which aspect of the machine learning process to be optimised for energy efficient and faster AI is being explored by various researchers. In most of the literatures preceding this dissertation, hyperparameter optimisation has been used to find optimal model hyperparameter configurations in an attempt to build faster and energy efficient AI. It is therefore of great essence that other aspects of the ML process other than tuning the hyperparameters be investigated as well to see how best they can be optimised in building energy efficient ML models with overall reduced harm to planet earth. Hyperparameter optimisation and feature optimisation are used in almost every AI/ML project. As earlier mentioned, researchers interested in measuring the energy consumption mostly focus on investigating the effects of hyperparameter optimisation only but this dissertation tries to investigate the stand-alone contributions of hyperparameter optimisation, feature optimisation and their combined effects as applicable techniques towards a faster and greener AI.

2.3 Hyperparameter Optimisation for Faster and Energy Efficient AI

State-of-the-art machine learning algorithms most especially supervised ML algorithms come with hyperparameters that can be tuned for optimal predictive performance. These hyperparameters can be applied in three approaches in ML projects; using default values as set by the developers of the algorithm, manually configuration the values and finally by using optimisation techniques to carry out the tuning procedure [46]. The later approach, is commonly referred to as hyperparameter optimisation. This dissertation makes use of the later approach, that is using optimisation techniques to carry out the configuration of the hyperparameters. This approach albeit is computationally expensive, remains the approach of choice in the quest for optimal predictive models. With this approach, it is possible to trade-off portion of accuracy so as to reduce the computational and environmental costs. It is therefore not strange that almost every researcher in the AI community interested in the area of faster and energy efficient models resort to hyperparameter optimisation. Although, hyperparameter optimisation is an everyday task in machine learning as it is primarily used for achieving higher predictive accuracies, investigations of its contribution to building energy efficient models at both extremes of model training and inference are now gaining prominence.

It is however worth noting, that not much has been done to fully help understand the impact of hyperparameter optimisation as a technique for building faster and energy efficient AI. This is backed by the fact that, there only exist a limited number of literatures on this all-important topic. In one of these few studies, the researchers explored the hyperparameter configurations for MLP algorithm by using grid search optimisation technique on five classification datasets. The research focused on establishing trade-off between classification accuracy and the level of energy consumed on training the model or on inference. They showed that 77% inference energy could be saved as evidenced in one of their datasets with a marginal loss of 1.1% in model accuracy. They also indicated that training energy could be saved by a margin of between 30% and 50%.

2.4 Feature Optimisation for Faster and Energy Efficient AI

Aside hyperparameter tuning, one of the aspects of the machine learning process which is widely known to reduce computational time and also believed to be helpful in building predictive models that achieve same or even better accuracy with reduced features of the dataset is feature selection. However, in machine learning projects, researchers have actually not tried to explore this technique as one of the means of cutting down the energy consumption used up by ML

algorithms at both ends of training and inference. This is backed also by the insufficient literature on estimation of the energy consumption levels to ascertain the extent to which feature selection makes AI/ML faster and environmentally friendly, albeit researchers believe FS has the capability.

Feature optimisation is the method of diminishing the number of input factors when creating a predictive model. It is alluring to decrease the number of input factors to both decrease the computational cost of modelling, in a few cases, to speed up the experimentation as well [47]. There are numerous distinctive feature optimisation algorithms, in spite of the fact that they can broadly be gathered into two primary categories: *filter* and *wrapper* methods. *Filter feature optimisation approaches* utilize statistical methods to assess the relationship between each input attribute and the target attribute, and these scores are utilized as the premise to select which features are appropriate to include in building the predictive model. *Wrapper feature optimisation* on the other hand, make numerous models with diverse subsets of input attributes and select those features that result within the best performing predictive model agreeing to an execution metric. These approaches do not rely on the type of variables, although they can seem to be computationally costly [48]. Recursive Feature Elimination with Cross Validation (RFECV) [49] may be a good case of a wrapper feature optimisation approach and this is what has been used in this dissertation.

These days, being in computerized time the information produced by different applications are expanding definitely both column-wise and row-wise; this makes a bottleneck for analytics conjointly increments the burden of machine learning computations that work to detect patterns. This cause of dimensionality can be dealt by employing techniques that reduce the dimensions. According to Venkatesh and Anuradha [50], who reviewed feature optimisation approaches, the authors concluded from their extensive study that most of the feature selection (FS) strategies utilize static data. In any case, after the rise of IoT and web-based applications, the information are produced powerfully and develop in a quick rate, so it is likely to have noisy data, which also can ruin the algorithm's performance. The use of feature optimisation methods not only reduce the dimensions of the dataset but also helps in avoiding overfitting the model as well.

One area where FS has featured prominently is its ability to cut down on the energy consumption of wearable sensory devices. Sensory devices which are wearable are getting to be the empowering innovation for numerous applications in well-being and healthcare, where computational components are firmly coupled with the human body to screen particular developments to the wearer. Algorithms mainly used for classification problems are the foremost commonly utilized ML algorithms that are capable of detecting which aspects of the system are of interest. The utilization of exact and resource-efficient classification algorithms is of key significance since wearable nodes work on restricted resources on one hand and proposed to recognize basic developments (e.g., falls) on the other hand. These calculations are utilized to outline factual highlights extricated from physiological signals onto diverse states such as wellbeing status of or to understand the movement performed by a subject. Ordinarily chosen highlights may lead to quick battery consumption, basically due to the nonattendance of computing complexity measure whereas selecting noticeable highlights. The researchers in [18] presented the idea of power-aware feature selection, which sought to reduce the energy taken up by these wearable sensory devices during processing of data for classification applications such as in recognition of actions. The authors approach takes into account the stand-alone computational cost of the features, computed in real time. The researchers introduced a graph model which is representative of the computing complexity and the correlation of the features. They used applied greedy search and integer programming approaches in deciding which features to include in a power efficient way. The researchers indicated upon experimentation over 30 channels task-based data collected that, their methodology is appropriate and very significant in its capacity to reduce energy consumption by 30%.

More so, one notable feat in the application of feature selection as a tool for reducing energy consumption and making the environment green is the work by Ding et al. [34]. The researchers were motivated by the limited literature on approaches towards reducing computational and power consumption while building wearable devices that are able to recognise human activities, commonly known as Human Activity Recognition (HAR) devices. The authors proposed and improved HAR system which is based on random forest to assist in caring for the elderly. Their proposed framework extricates three sorts of pairwise relationship features in crossover sliding windows, and based on location data to improve the performance in recognition. A shared data feature selection is received to optimize the recognition of confounded local set of tasks. The researcher's system, made it possible for the number of trees to be reduced while maintaining the level of recognition accuracy. They found that, their approach could predict 10 classes of tasks with 93.01 percentage accuracy and 74.9 percentage savings in energy consumption. Although the researchers contributed significantly to general knowledge of feature selection as a tool to cut down on energy consumption and even to improve accuracy, the study only used random forest algorithm. Using other algorithms would help to generalise this further. This dissertation applies feature selection by using the XGB classification algorithm although in a different fashion to complement the findings of Ding et al. [34].

2.5 A Blend of Hyperparameter Optimisation and Feature Optimisation for Faster and Greener AI

Following from sections 2.3 and 2.4, the significance of both hyperparameter optimisation and feature optimisation as techniques to help reduce the ever-increasing energy consumption of ML algorithms cannot be relegated. Hyperparameter optimisation on its own has been useful in building energy efficient AI/ML models as shown by Brownlee et al. [1] among few other researchers. On the other part, feature optimisation on its own has been illustrated by few researchers such as [18], [34] to be an effective approach in reducing energy consumption. While feature optimisation may lead to a little drop in model's predictive accuracy, a substantial reduction in the energy taken up by the algorithm is preferred for a faster and greener AI. The search for literature indicated that an attempt to measure the combined impact of hyperparameter optimisation and feature optimisation on energy consumption has not been fully explored. Although the authors of [51] treated feature optimisation and hyperparameter optimisation in a multi-objective task, their focus was on how simultaneously tuning the hyperparameters and performing feature selection may impact on predictive performance. The researchers made no attempts to estimate how fast the model runs or energy is saved by the approaches they used. An attempt to explore faster and energy efficient AI/ML algorithms by using hyperparameter optimisation as applied on selected features while maintaining the predictive accuracy of the model is therefore very significant.

2.6 Critical Review of Directly Related Work

In this subsection of the dissertation, a critical review is expressed on the work of Brownlee et al. [1] titled "Exploring the Accuracy – Energy Trade-off in Machine Learning" and published in 2021.

Not much has been done by researchers to help the AI community on approaches to faster and energy efficient AI while maintaining accuracy, although power-awareness has been created in the AI community in recent times. The work by Brownlee et al. [1], follows from the research work of Brownlee et al. [13]. Whereas [13]'s experimental works were done on a Java environment, [1] carried out the experimentations on a python environment which is a very significant contribution to the AI community as the shift towards the python programming environment is increasingly gaining popularity in AI.

The topic chosen for the paper is very clear and appropriate based on the content. One of the key strengths of the research under review is the purpose for which the research was undertaken. The authors stated that, their main motive of the research undertaken was to investigate further the chances of hyperparameter optimisation technique on the need for developing energy efficient models. The authors formulated their research questions excellently as this helped them to breakdown the broader topic into smaller and specific objectives which were realistic and achievable. One strength of their research has to do with the methodology and tools used. The authors outlined them in such a way that it is easier for one to replicate their study. However, with the exception of the mortgage lending dataset, the number of instances in the other datasets following from which generalisations were made do not actually mimic the number of real-life dataset instances. The need for additional or new datasets to complement their experimentation and findings would therefore help in the generalisation further. Another major strength of [1] worth mentioning is the measurement of the energy consumption. Unlike other researchers such as Noureddine et al. [52], who used CPU time which has been proven to be inaccurate by Zhang et al. [27] for the fact that it ignores CPU idle time as proxy to measure energy consumption, Brownlee et al. [1] used Intel's CPU based RAPL to measure the energy consumption.

One main criticism of the research also has to do with the train-test split proportion used. Would their results have been same if another train-test split ratio was used? Also, with regards to the MLP machine learning algorithm used, would they have achieved same results in case they used a different machine learning algorithm other than the MLP? One of the issues of concern with the research is that, the authors were silent on the runtime of the models they built although just a little was mentioned in their reporting of the results. Much emphasis was placed on accuracy and energy efficiency. It could be inferred however, that an energy-efficient model would certainly use less computational runtime, but using the experimental results to confirm and bring that knowledge to the fore would have helped the AI community. Although the study by Brownlee et al. [1] is excellent, the researchers failed to let readers know what the actual energy consumptions were before they applied the optimisation techniques. This could have been improved in a consistent series of experimentations.

Although the need for a greener AI is gradually seeping into the interests of players in the AI community, much research has not been done about this all-important topic, as evidenced in the just a few availabilities of previous literature. This makes it a daunting task to research on the topic but motivated by the work of Brownlee et al. [1], this dissertation will investigate hyperparameter optimisation and further explore feature optimisation as techniques to employ in the quest of achieving a faster and greener AI with minimal or no adverse consequence on predictive accuracy.

3 Methodology

This section of the research work outlines the approach used in achieving the set objectives. The section is presented in the following subsections; research design, dataset, the general approach, measurements and finally, technology and hardware systems used to carry out the experiments.

3.1 Research Design

The primary aim of this research is to make ML/AI modelled systems faster and greener. Based on existing literature, hyperparameter optimisation and feature optimisation were chosen as the techniques to investigate and gain understanding of how they contribute towards achieving the aim. In this regard, the computational runtime and energy consumption of XGB classification algorithm was applied on the SGS dataset in series of four experimentations, see *Figure 3*. The following subsections relate to the series of experimentations that were carried out to help in establishing the facts of the research objectives.

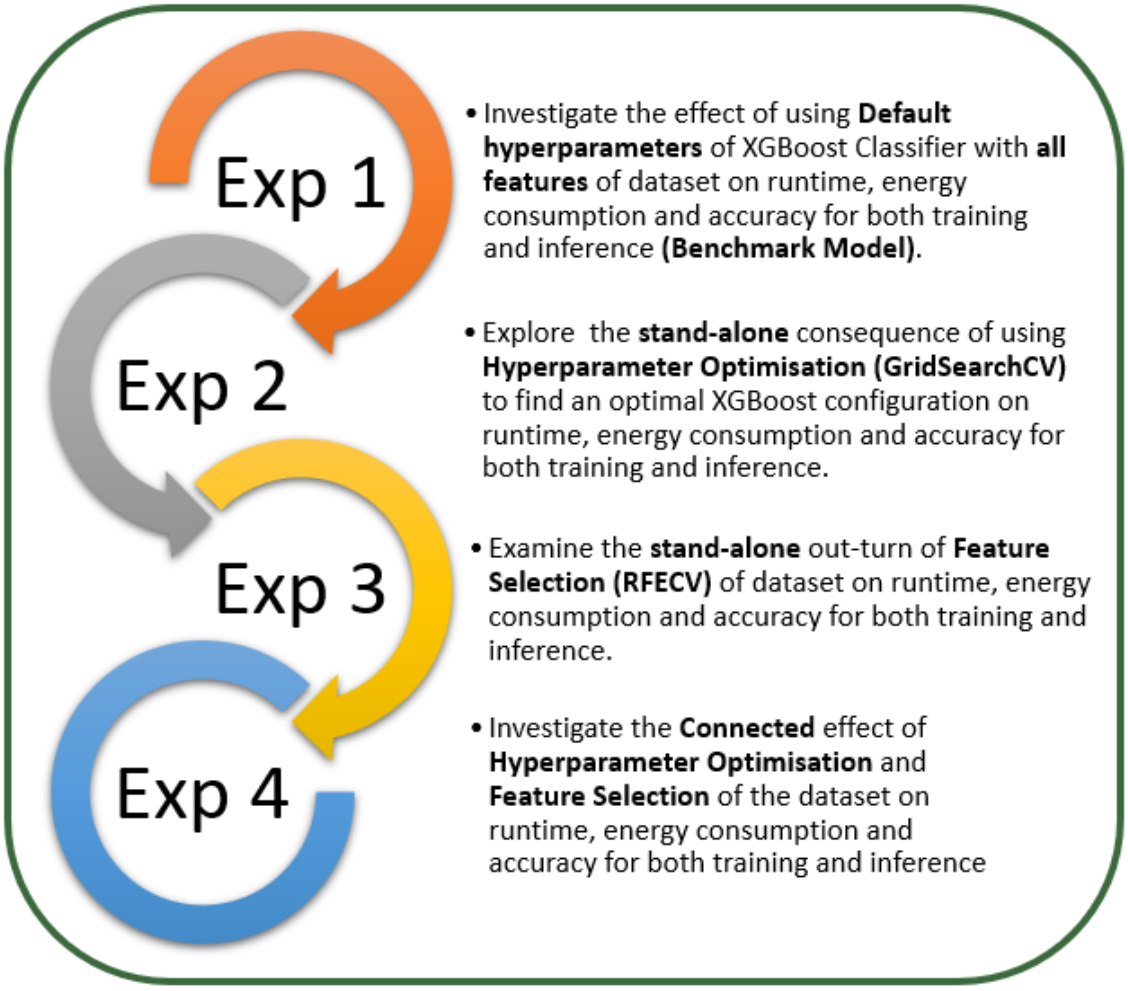

Figure 3. The Experimental Series

3.1.1 Benchmark Model (Experiment 1)

To appreciate the problem and also to have a standard of measure for the other series of experimentations, a benchmark model (experiment 1) was designed. This model was constructed by using the default hyperparameters of the XGB classification algorithm on the entire features of the SGS dataset and measuring the runtime and energy consumption at both extremes of training and inference. This was very informative as it helped in directing focus on to the phase of the ML cycle that takes longer to compute and also takes up much energy and needed a remedy to keep the environment green.

3.1.2 Stand-alone Hyperparameter Optimised Model (Experiment 2)

Experiment 2 explored the stand-alone effect of using hyperparameter optimisation as a technique in making AI faster and greener. Grid search specifically *GridSearchCV* provided by scikit-learn for python was used to optimise the hyperparameters of the XGB classification algorithm on an arbitrary search space as shown in *Table 1* below. The choice of the values in the search space was motivated by the need to get a good spread around the default hyperparameters of the XGB classification algorithm as provided by scikit-learn. It is significant to note that, all the entire features of the SGS dataset were used in this experimentation (experiment 2) to construct the model.

XGB Hyperparameter	Values used in the space	Description
colsample_bytree	0.1, 0.3, 0.5	Subsample ratio of columns when constructing each tree
learning_rate	0.05, 0.1, 0.2	A tree booster step size shrinker used in contracting weights of the features just so the boosting process is made more conservative
max_depth	2, 3, 5	The maximum depth of a tree
min_child_weight	2, 3, 5	Minimum of instance weight needed in a child
n_estimators	50, 80, 100	Number of gradient boosted trees
subsample	0.3, 0.5, 1	Subsample ratio of the training instances

Table 1. XGB Classifier Hyperparameters and the Search Space

3.1.3 Stand-alone Selected Features Optimised Model (Experiment 3)

In the quest of understanding the stand-alone impact of feature optimisation as a technique in making AI faster and greener, this experimentation was necessary. Recursive Feature Elimination

with cross Validation (RFECV) was employed to trim down the number of features that were used in building the model by ensuring that only features of importance are included in the model. It is worth mentioning that the default hyperparameters of the XGB classification algorithm were used on the selected features of relevance as revealed by the feature optimisation technique in experiment 3.

3.1.4 Tuned Hyperparameters on Selected Features Optimised Model (Experiment 4)

This experimentation is the core of the research as it seeks to investigate the connected effect of hyperparameter optimisation and feature optimisation as techniques in helping to make AI faster and greener. In this final part of the series of experimentations, the hyperparameters of the XGB classification algorithm were tuned on the selected features of the SGS dataset. What this means is that, *RFECV* was used to select the features of relevance that would yield optimum output after which *GridSearchCV* was used over the search space as in *Table 1* above to get the best configuration of the hyperparameters that will make the model to run faster, consume less energy and with marginal or no loss in the predictive accuracy of the model being constructed.

3.2 Dataset

Five datasets were initially under consideration for this project work. Due to the limited timeframe, this research was conducted with one dataset which is of great importance in the application of AI/ML technology. The Smart Grid Stability [19], [53], [54] dataset was specifically chosen for this research. Smart grids have many energy-hungry interlocking features intelligently working together and key among them is the communication and control systems which have great impact on grid stability. The need to therefore cut down on the energy consumption that comes with monitoring the stability of the smart grid therefore served as a huge motivating factor for the choice of the SGS dataset in favour of the others which were under consideration.

3.2.1 Dataset Source and Description

The Smart Grid Stability dataset is freely available on Kaggle [19]. It is an augmented version of the "Electrical Grid Stability Simulated Dataset" hosted by the University of California (UCI) Machine Learning Repository.

The SGS dataset is formatted as a comma separated value (csv) file and hosted on Kaggle with the name smart_grid_stability_augmented.csv. The dataset as used in this project is synthetic in nature and consists of results taken from grid stability simulations performed on a four-node network with star architecture *Figure 4*, and as explained in *Table 2*.

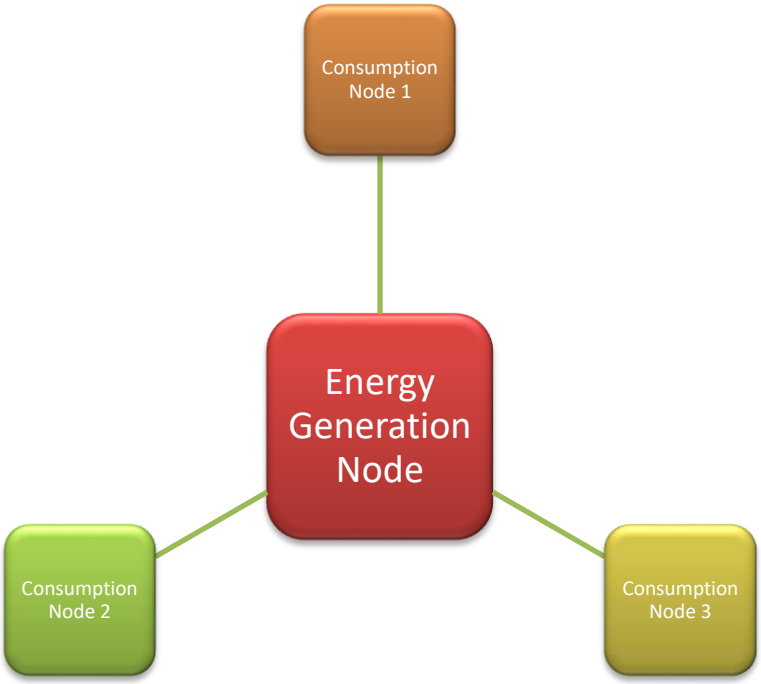

Figure 4. Four-Node Star Topology of Smart Grid

The dataset comprises of 60,000 instances and 14 attributes (See Appendix 3 for the first five and last five instances of the data as originally made available on Kaggle). Out of the 14 attributes, 12 of them are predictive feature attributes and the remaining 2 are target attributes. Table *2* shows and explains the 14 attributes of the SGS dataset used in this research.

Feature		**Data Type**	**Explanation**
1	tau1	Numeric – Continuous	These attributes represent the reaction time of each network participant with a real value range from 0.5 to 10. Whereas *tau1* represents the energy generation node, *tau2* – *tau4* represent the consumption node participants in the architecture
2	tau2	Numeric – Continuous	
3	tau3	Numeric – Continuous	
4	tau4	Numeric – Continuous	
5	p1	Numeric – Continuous	*p1* to *p4* indicate the nominal power produced (positive) or consumed (negative) by each network participant, a real value within the range − 2.0 to − 0.5 for consumers (*p2* to *p4*). As the total power consumed equals the total power generated, *p1* (supplier node) = (*p2* + *p3* + *p4*)
6	p2	Numeric – Continuous	
7	p3	Numeric – Continuous	
8	p4	Numeric – Continuous	

9	g1	Numeric – Continuous	*g1* to *g4* indicate the price elasticity coefficient for each network participant, a real value within the range 0.05 to 1.00 (*g1* corresponds to the supplier node, *g2* to *g4* also correspond to the consumer nodes; *g* stands for 'gamma')
10	g2	Numeric – Continuous	
11	g3	Numeric – Continuous	
12	g4	Numeric – Continuous	
Target		**Data Type**	**Explanation**
1	stab	Numeric – Continuous	*stab* represents the maximum real part of the characteristic differential equation root (if positive, the system is linearly unstable; if negative, the system is linearly stable)
2	stabf	Nominal – Categorical	*Stabf is* a categorical (binary) label ('stable' or 'unstable'). This was reengineered from the *stab* attribute

Table 2. Description of Dataset

3.3 General Approach

This subsection outlines the general machine learning approach adopted in the series of experimentations.

3.3.1 Data Preparation and Pre-processing

In most instances of machine learning projects, the datasets used do not come in a form appropriate enough to assist in building trustworthy models; in this regard, data preparation and pre-processing are key as they allow us to get the data in a format suitable for building models that lead to accurate insights.

The SGS dataset is generally clean in nature and does not require any further cleaning before usage in projects. Thus, the SGS dataset does not suffer from the problem of missing data, unwanted outliers, structural errors, irrelevant observations and other such issues associated with raw datasets. However, deeper understanding of the dataset led to the choice of which target attribute to use in this research. As explained in *Table 2* above, the target attribute *stabf* was engineered from the *stab* attribute indicating a direct relationship between these two target attributes. The *stab* target attribute was therefore dropped leaving *stabf* as the only target attribute for this study. The numerical nature of the dataset coupled with its clean nature informed the immediate start of modelling skipping the pre-processing step.

In the preliminary stages of this research, correlation test was performed to ascertain the relationship between the target attribute and the feature attributes and also amongst the feature attributes. No correlation was observed between the feature attributes and the target attribute. Also, no correlation was found among the features. A strong correlation in excess of -0.80 was found between *stab* and *stabf* and this supports the decision made to drop *stab* in favour of *stabf* as the only target attribute for this research work. *Figure 5* depicts the heatmap of the correlation matrix of the SGS dataset attributes.

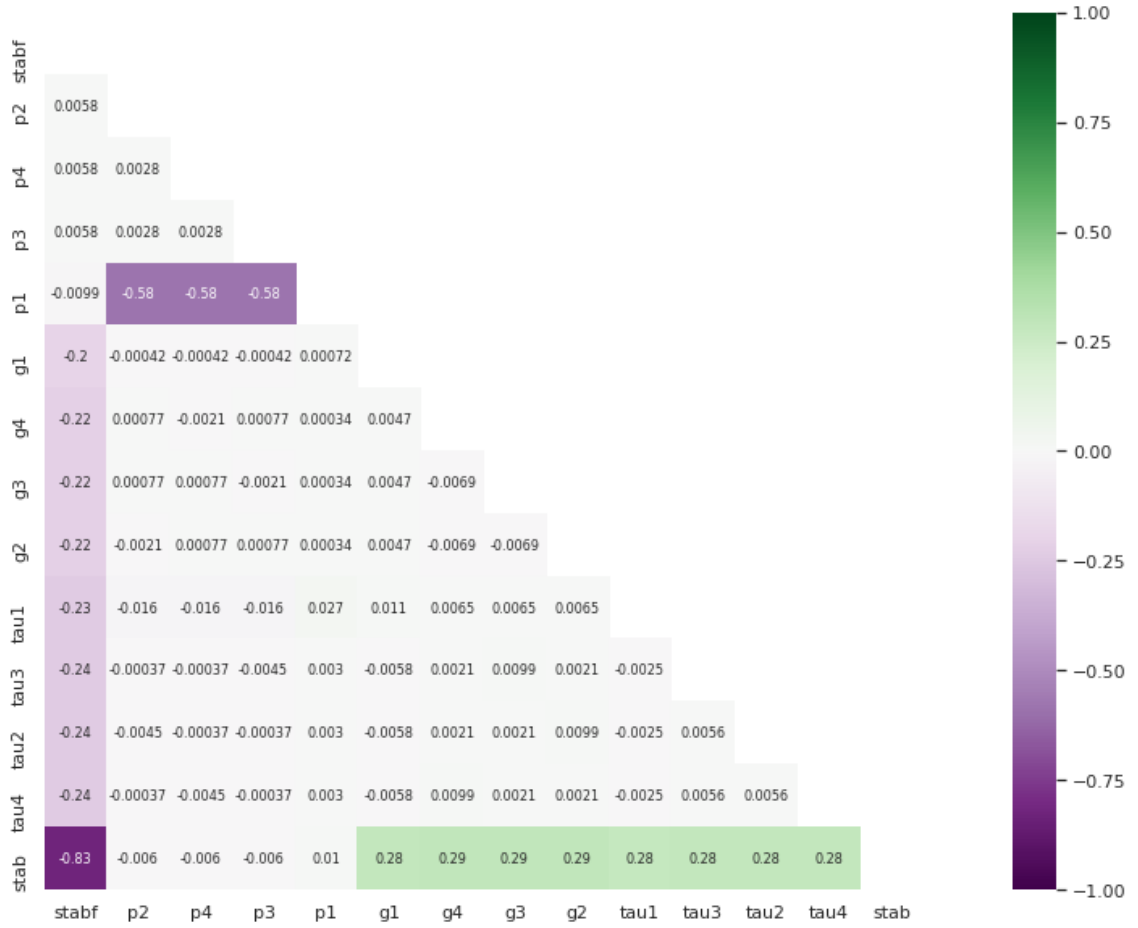

Figure 5. Heatmap of the Correlation matrix of SGS Dataset Attributes

3.3.2 The General Train-Test Approach

Unless otherwise stated in a specific experimental stage, the general approach below outlines the machine learning steps applied in all the 4 series of experimentations involved in this research work.

To begin with building machine learning models, it is of utmost importance that the dataset be split into 2; one part for training the model and the other for testing the model. Following from literature and as a good practice, the 80:20 split was adopted for this research. The 80%-part split of the SGS dataset was used in training and validation of the model whereas the remaining unseen and untouched 20% was used for inference purposes.

Since the dataset is most appropriate for classification tasks, during the early stages of the project, various classification machine learning algorithms such as MLP and Random Forest among other were tried on the SGS dataset. As the focus of this work is to make AI faster and greener and without necessarily causing much harm to the predictive accuracy of the model, the *XGB classifier* was selected as the algorithm for this study. It outperformed the other classifiers by achieving the highest accuracy in its default state. However, for the purpose of this research, *hyperparameter optimisation* was used to fine-tune the *XGB classifier* in experiments 2 and 4 as earlier described in the research design section.

XGB classifier was fitted to the training dataset to train the model. For evaluation of the models performance, cross validation with k-fold was used. The model training was repeated 30 times and the medians of *CPU duration*, *runtime duration*, *CPU energy consumption*, *DRAM energy*

consumption and *Cross validation accuracy* were recorded. This generic process was performed in each of the 4 experiments of this research.

The trained model was then tested on the 20% unseen and untouched test dataset to assess the constructed model's predictive accuracy. Since inference in machine learning projects are extremely faster as compared to training, the testing process was repeated 1000 times at this stage instead of the 30 times performed for the training stage. The medians of *CPU duration*, *runtime duration*, *CPU energy consumption*, *DRAM energy consumption* and *Test accuracy* for the 1000 repeated runs were recorded. This generic approach was followed in each of the 4 experimental series of this work. Considering the non-parametric nature of the results, *Mann–Whitney U test* was then performed to establish the statictical variations in the various results especially with focus on experiment 1, the benchmark results and experiment 4, the core objective results.

3.4 Measurement of Performance

With the research aim of making AI faster and greener and as a matter of standard practice, it was appropriate that some measurement metrics are put in place as a guide towards the attainment of the specific objectives of this research. These metrics for the purpose of this research in no particular order are classed into three; Evaluation and test accuracy metrics, Time and energy consumption measurement and Statistical metrics.

3.4.1 Evaluation and Test Accuracy Metrics

It is important to ascertain how best or poorly a machine learning algorithm or system performs in real-life applications prior to deployment. In this respect, it is key that certain metrics are used to assess the performance of the model built and then tested on an unseen data as well as though it was being used to tackle real-life problems. For the purposes of this research work, the mean accuracy over 5-folds of cross validation as measured by *k-fold Cross Validation scored with Accuracy* is used to evaluate the performance of the fitted model on training, whereas *Accuracy* is used to measure how best the constructed model is able to predict on unseen and untouched data during inference.

3.4.2 Time and Energy Consumption Measurement

Since the research in its entirety was inspired by the work of Brownlee et al [1], measurements of time and energy consumption followed the same approach. Python's *time* library was used to measure the *CPU time* as well as the *runtime* in seconds. This only helps to acknowledge how fast the model runs and it is not in any way a measure of energy consumption as proxied in other works. Also with the energy consumption, *pyRAPL* library which is based on Intel's *"Running Average Power Limit" (RAPL)* technology was used to estimate the energy consumption of the CPU when a piece of machine learning task is executed. The technology has been in existence since the *Sandy Bridge* generation. pyRAPL can more specifically measure the energy consumption of the CPU's socket package domain, DRAM (server architectures) domain and GPU (client architecture) domain. The energy is measured in *microjoule (μJ)* equivalent to one millionth (10^{-6}) of one *joule*. For the purposes of this work, the energy measured was converted to *Joules* (See sample code of pyRAPL usage in Appendix 3). *Table 3.* below gives a summary of the time and energy consumption metrics employed in this dissertation.

Metric	Unit	Description
CPU Duration	Seconds (S) with nanosecond resolution	• The time taken by the CPU to fit the model to the training data if it relates to the training phase as reported by python's *time.process_time()* library. In this research this is referred to as *Training CPU Duration.* • The time taken by the CPU as reported by python's *time.process_time()* method for the application of the model to test on unseen and untouched data, this is termed as *Inference CPU Duration* in this research.
Runtime Duration	Seconds (S)	• The wall-clock time equivalent taken to fit the model to the training data, as reported by python's *time.time()* library. In this case, this is referred to as *Training Runtime Duration*. • The wall-clock time equivalent taken to apply the model to test on unseen and untouched data as reported by python's time.time() method. In this dissertation this is referred to as *Inference Runtime Duration*.
CPU Energy Consumption	microjoule (µJ) x (10^{-6}) = Joules (J)	• Energy used by the CPU as reported by *pyRAPL* for model fitting if it relates to training. For the purposes of this dissertation, this is termed as *Training CPU Energy Consumption.* • Energy used by the CPU as reported by *pyRAPL* for the application of the model to test on unseen and untouched data. For the purposes of this dissertation, this is termed as *Inference CPU Energy Consumption.*
DRAM Energy Consumption	microjoule (µJ) x (10^{-6}) = Joules (J)	• Energy used by the DRAM as reported by *pyRAPL* for model fitting if it relates to training. For the purposes of this

		dissertation, this is termed as *Training CPU Energy Consumption.* • Energy used by the DRAM as reported by *pyRAPL* for the application of the model to test on unseen and untouched data. For the purposes of this dissertation, this is termed as *Inference CPU Energy Consumption*

Table 3. Description of the Time and Energy Consumption Measures

3.4.3 Statistical Metrics

The *Mann-Whitney U test* is used to compare whether there is a difference in the dependent variable for two independent groups. In this dissertation, the Mann-Whitney U test was used to ascertain if the distribution of the dependent variable is the same for the two groups and therefore from the same population. Each of the energy consumption results for the experiments 2,3 and 4 were compared to experiment 1 (the benchmark model) for both training and inference (See sections 3.1.1 – 3.1.4). This was done to be sure of the statistical significance of the experimental differences in the medians of results among the various experiments conducted. More so, the runtime scores of experiments 2,3 and 4 were compared to that of experiment 1 at both extremes of training and inference to ascertain, if the experimental differences in their medians were statistically significant. The use of the Mann-Whitney U test was appropriate for this purpose since it makes no assumption of normality. In setting up the Mann-Whitney U test to investigate if the experimental medians were statistically significantly different, the following null (H_0) and alternate (H_A) hypotheses were formulated and made applicable for both energy consumption and runtime at both ends of training and inference;

H_0: The distributions from Exp1 and Exp(*n*) are equal

H_A: The distributions from Exp1 and Exp(*n*) are not equal

Where *n* denotes either of experiments 2, 3 or 4.

In setting the rejection or acceptance criteria, an *alpha (α)* value of 0.05 which is representative of a 95% confidence interval was used. The rejection or acceptance criteria was then formulated for both energy consumption and runtime during training and inference as follows;

If (ρ-value > α), Accept H_0 and Reject H_A

Else if, (ρ-value < α), Reject H_0 and Accept H_A

What the above criteria means is that, the experimental medians are equal and therefore statistically insignificant when the ρ-value is greater than 0.05. the conclusion from this therefore is that, the null hypothesis is to be accepted. The other side of the criteria above also is an indication to mean that, when the ρ-value of the experimental results is less than 0.05, it is concluded that the differences in median of the two experiments compared are statistically significantly different and therefore the alternate hypothesis is accepted.

In addition to the use of the Mann-Whitney U test to determine the statistical significance of the medians of the series of experimentations conducted, *violin plots* were also employed to give visual representations of the distributions between the series of experimentations. Although, the median, interquartile range and other such measures are good statistical measures but using

them alone do not help to get enough understanding of the results as it fails to give the distributions. This is why distribution plots come in handy. The box plot could have been used but it is limited by its inability to in visualising variations in the dataset. This therefore informed the use of the violin plots. The violin plot is a blend of kernel density plot and box plot and it is able to indicate peaks in the dataset.

3.5 Technology and Systems used

The programming language and environment, main software packages used as well as the specifications of the system hardware used in carrying out the experiments are briefly described in this subsection.

3.5.1 Software Used

Python 3.8.8 programming language running in *Jupyter Notebook 6.3.0* on *Anaconda 2.0.3* was used to carry out the research. Python was specifically chosen for this project due to its high level of readability and maintenance which has made it a popular language among developers. Again, the choice of python over the other programming languages for this project was motivated by its advantage being able to run on a wide variety of hardware platforms with almost the same interface and also its broad standard library compatibility on all of these different platforms.

Scikit-learn 0.24.1 was one of the key python libraries used to perform the series of experiments. It is a free to use machine learning library within the python programming language. It comes with several algorithms that are useful for classification, clustering and regression among many other machine learning purposes. The tools used in data preparation and pre-processing relied greatly on the scikit-learn library. Again, the *GridSearchCV* and *RFECV* optimisation techniques used for the hyperparameter configuration and feature selection respectively were all implemented from the scikit library. More so, the metrics used in evaluating the model's fit and also for testing accuracy relied on this library. The choice of scikit-learn over the other python libraries for this project work was informed by the spread of machine learning algorithms coupled with the tools that comes with it such as the cross-validation tool for evaluation of accuracy of models on unseen data which has made its usage popular among developers. It is important to note as early mentioned that most of the important tools required to carry out the project were already available in the scikit-learn library.

The *eXtreme Gradient Boosting (XGB)* classification algorithm implemented in this experiment was provided by the XGBoost library in python. This is a prebuilt binary wheel available from Python Package Index (PyPI). The XGB is a tree boosting algorithm that acts on shallow and weak trees by converting them into strong learners using weighted averages. It is designed to compute faster and also for optimised performance as well. This is a key motivation for its implementation in this project work over the other machine learning algorithms. It is now popular in usage among various players in the AI community due to its ability to deal with sparse data and its ability to run in parallel as compared to other tree boosting machine learning algorithms. The choice of XGB over classification algorithm in this project was also in part motivated by its proven performance on classification tasks as used in both industry and winning competitions. Furthermore, its compatibility and portability permit usage on almost all the major platforms such as Linux, Windows and the OS X leading to its usage in this project over the other machine learning algorithms such as Random Forest, K-Nearest Neighbours and the likes.

pyRAPL 0.2.3.1 was implemented to estimate the energy consumption. This library provides a python interface to Intel's "Running Average Power Limit" (RAPL) technology. This technology provides energy consumption estimates of all CPU sockets as well as the DRAM sockets. pyRAPL has limited popularity in usage based on project statistics from the GitHub repository as well as

its maintenance being inactive since December 2019. In spite of this limitation, pyRAPL has proven to be better estimator of CPU energy consumption than the CPU time as used by other researchers (See section 2.6). This motivated the choice of pyRAPL as the tool to measure the energy consumption in this project.

The *time* library was also used to estimate the wall clock time equivalent a process took to run. Other libraries such as pandas 1.2.4, numpy 1.19.5 and scipy were also used in the experimentation based on their proven roles in data science projects.

3.5.2 System Hardware Used

The experiments embodied in this dissertation were performed on a workstation running Debian OS, with 12-core (24 with hyperthreading) Intel Xeon 2620 V3 with each running at 2.4GHz and 15M cache Skt2011 processor. Also, the workstation used in this project had a total of 32GB (4 x 8GB) DDR4 2133MHz ECC registered memory. Other components of the workstation include a 250GB 6Gb/s 2.5” Solid State disk, 2 x Hitachi 2TB 7200rpm 6Gb/s Enterprise SATAIII disk drive and nVidia TitanX GPGPU. It is important to note that no other computationally intensive processes were run concurrently and also the experiments were run sequentially.

4 Results and Discussion

This section of the dissertation presents the findings made from the series of experimentations performed. It also seeks to discuss the findings based on other previous works in the literature. The order of the presentation is in line with the objectives set for this dissertation. What this means is that, the results is presented based on the four thematic topics of this dissertation which include; the benchmark model which estimates the time and energy consumption to actuate the environmental cost of AI, hyperparameter optimisation as a technique in building energy efficient AI models, feature optimisation as a technique in building energy efficient models and also a blend of hyperparameter optimisation and feature selection optimisation for building environmentally friendly AI models. The presentation follows with a general overall summary of the findings and a general discussion on the move towards a faster and greener AI.

4.1 Results of the Benchmark Model

The benchmark model was setup to be the baseline experiment. As earlier mentioned in the methodology (See section 3.1.1), the purpose of the benchmark model was to actually help in understanding the environmental cost that comes with AI/ML algorithms in their default states. Furthermore, these results will subsequently be the standard of measure of the effects of hyperparameter optimisation and feature optimisation as techniques to building faster and energy efficient models. The results in this sub-section relates to experiment 1 where the default hyperparameters of the XGB classification algorithm were used in building the model on the entire features of the SGS dataset. The results are expanded based on the phase of the machine learning process, that is, either at the training or inference stage.

4.1.1 Training Results– Default hyperparameters on Entire Dataset Features

Inspection of the runtime and energy consumed in fitting the XGB classification algorithm with its default hyperparameter configurations to the entire features of the SGS after 30 repeated runs, stemming from experiment 1 resulted in the medians of the 30 repeated runs reported as shown in *Table 4* below.

Metric	Recorded
Training CPU Duration (S)	252.21
Training Runtime Duration (S)	11.78
Training CPU Energy Consumption (J)	1170.73
Training DRAM Energy Consumption (J)	57.71
Mean of 5-fold CV Accuracy (%)	97.80

Table 4. Experiment 1 – Training Results

From Table 4 above, it is noted that, without hyperparameter tuning of the XGB classification algorithm and reduction in number of dataset features used, it took approximately 11.78 seconds for the model to be fitted to the training dataset. On evaluation of the model, 97.8% accuracy was realised to indicate that the XGB classifier model has very high predictive power on the SGS dataset. However, the energy consumed by the CPU for the model fitting to the training dataset to be completed was recorded by pyRAPL as 1,170.73 Joules. Also, the CPU duration which in other research were used as proxy for energy consumption was recorded as 252.21 seconds and the energy taken up by DRAM as 57.71 Joules. The CPU energy consumption is

proxied in the context of this dissertation to indicate the environmental cost associated with training the XGB classifier algorithm on the SGS dataset whereas the runtime is proxied for how fast the model takes to be fitted. The results above presented answers in part, research question 1 and the attainment of objective 1.

4.1.2 Inference Results – Default hyperparameters on Entire Dataset Features

To fully answer research question 1 and completely achieve objective 1, the results of the inference stage of experiment 1 is presented in Table 5 below.

Metric	Recorded
Inference CPU Duration (S)	0.24
Inference Runtime Duration (S)	0.03
Inference CPU Energy Consumption (J)	1.67
Inference DRAM Energy Consumption (J)	0.10
Test Accuracy (%)	97.99

Table 5. Experiment 1 – Inference Results

As shown in Table 5 above, it took 0.03 seconds which is the median of the 1000 repeated runs for the model to be applied to test on unseen data. The test accuracy recorded was 97.99% indicating how robust the model is able to perform on unseen data. The CPU energy consumption which is indicative of the environmental cost in the context of this research was recorded by *pyRAPL* as 1.67 Joules. Other metrics such as the CPU duration and DRAM energy consumed on inference were also recorded as 0.24 and 0.10 respectively.

The results in this subsection augment the results in subsection 4.11, to fully address research question 1. It is evident from the experimental results in Tables 4 and 5 that, training a model takes a longer time to complete whereas using the trained model to test on unseen data completes extremely in a shorter period of time. In this case, it took *393times* and *701times* of the inference runtime and inference CPU energy consumed to train the model. This observation can be explained by the size of the training data as a result of the test-split ratio. The ML algorithm will have to learn from every instance in the train set before it is able to make decisions. Also, the ML algorithm used could account for the longer time difference as well as the higher energy consumed. Whereas some algorithms such as the K-Nearest Neighbours is a lazy learner and capable of learning in a very short time but takes long to execute, same cannot be said for the XGB classification algorithm. Following from the results above in *Tables 4 and 5*, it is demonstrated that it is environmentally costly to train a model than to use it in making inference.

The results in Tables 4 and 5 were used as the basis of comparison at both extremes of training and inference in subsequent experimental results with focus on training runtime and special focus on training CPU energy consumption which is used as an estimate of the environmental cost associated with fitting a model to a training dataset which has been demonstrated to be computationally and environmentally costly in the context of this dissertation.

4.2 Stand-alone Hyperparameter Optimised Model

The results in this subsection address research question 2 as well as the attainment of objective 2. Hyperparameter optimisation was explored to examine how it can be useful as a technique to reduce the runtime and energy consumption while trying to maintain accuracy at both ends of model training and inference. The results in this subsection relates to experiment 2 where *GridSearchCV* provided by scikit-learn was used to perform an exhaustive search on the finite

search space (as shown in *Table 1)*, around the default hyperparameter configuration of the XGB classification algorithm. The results of the best hyperparameter configuration based on experiment 2 that helps to achieve the broader aim of faster and greener AI while keeping accuracy at bay is presented in Table 6 below.

XGB Hyperparameter	**Best Configuration Value**
Colsample_bytree	0.5
Learning_rate	0.2
Max_depth	5
Min_child_weight	2
N_estimators	100
subsample	1

Table 6. Experiment 2 – Best Hyperparameter Configuration

The resulting runtime and energy consumption for both training and inference are presented in the following subsections.

4.2.1 Training Results – Tuned hyperparameters on Entire Dataset Features

The table below, Table 7 presents the results of using the optimised hyperparameter configuration on the entire dataset. This table is helpful in answering the training and energy consumption bit of research question 2. It also has a relative change column which represents the extent of the impact of the application of hyperparameter optimisation as a technique in constructing models that are not only accurate but also faster and greener.

Metric	**Recorded**	**Relative Change**
Training CPU Duration (S)	179.29	-28.91%
Training Runtime Duration (S)	8.67	-26.40%
Training CPU Energy Consumption (J)	819.68	-29.99%
Training DRAM Energy Consumption (J)	42.27	-26.75%
Mean of 5-fold CV Accuracy (%)	96.10	-1.74%

Table 7. Experiment 2 – Training Results

As noted from Table 7 above, with a trade-off of 1.74% model accuracy, training runtime was expedited by 26.4% while CPU energy savings of approximately 30% was achieved on training. Also, other metrics recorded, such as CPU duration and DRAM energy consumption were all reduced by 28.91% and 26.75% respectively relative to experiment 1 where no hyperparameter optimisation was performed. What this result means is that hyperparameter optimisation is useful as a technique to fall on in attempting to develop accurate, faster and most importantly eco-friendly AI models. As energy consumption is of utmost concern and noting from above experiment 1 that much energy is consumed during training, *Figure 6* below will throw more light on the relative CPU energy consumption between experiment 1 and experiment 2. This figure below

is useful in understanding how the hyperparameter optimisation faired in all the 30 repeated runs as compared to experiment 1 where no hyperparameter configuration was performed.

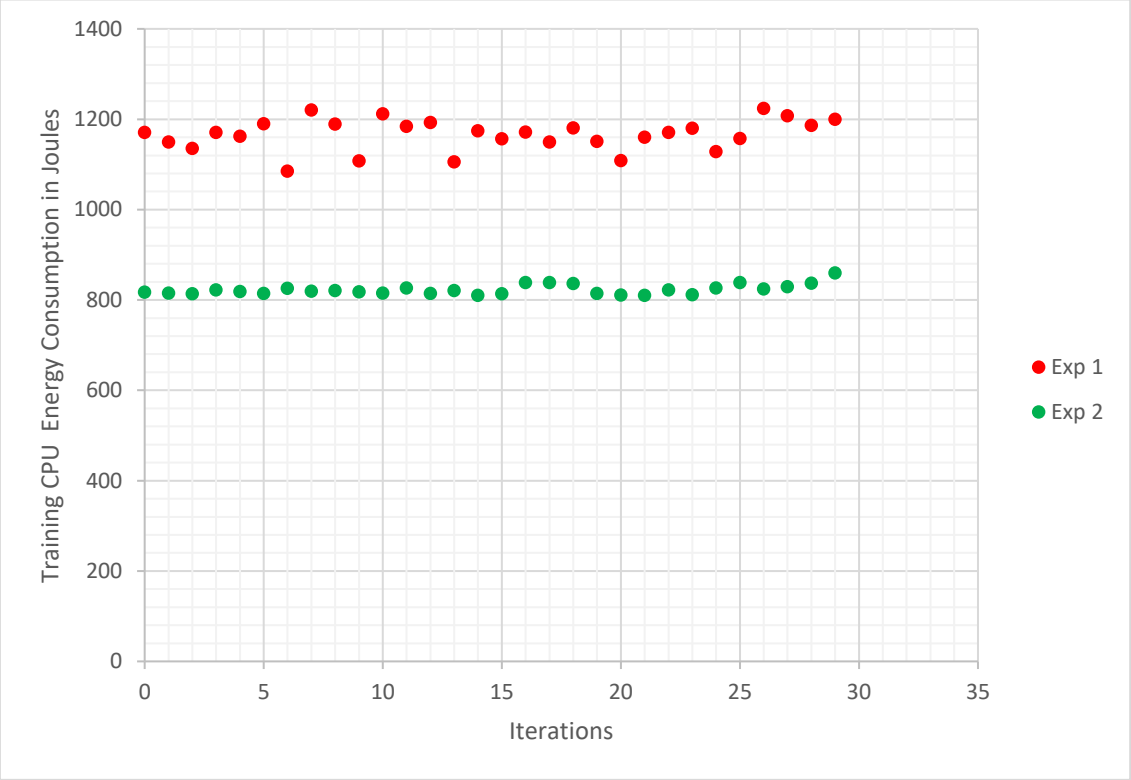

Figure 6. Training CPU Energy of Experiments 1 and 2 Compared for 30 Iterations

It can be inferred from the above figure that none of the 30 repeated runs of the training CPU energy consumption in experiment 2 exceeded or had the gap closing up to that of experiment 1, where no hyperparameter optimisation was performed. This is a strong indication that hyperparameter optimisation is a key technique to apply in attempting to build faster and greener AI models with minimal or no disturbance to predictive accuracy. This finding is a confirmation of the work of Brownlee et. al [1] who also found that training energy can be cut down by between 30% t0 50% with a minimal loss in predictive accuracy by applying hyperparameter optimisation.

4.2.2 Inference Results – Tuned hyperparameters on Entire Dataset Features

To complement subsection 4.2.1 in addressing research question 2 and for the complete achievement of objective 2, *Table 8* below presents the findings of the impact of hyperparameter optimisation on the entire features of the dataset under consideration at the inference phase of experiment 2.

Metric	Recorded	Relative Change
Inference CPU Duration (S)	0.22	-8.33%
Inference Runtime Duration (S)	0.026	-13.33%
Inference CPU Energy Consumption (J)	1.61	-3.59%
Inference DRAM Energy Consumption (J)	0.097	-3.00%
Test Accuracy (%)	96.27	-1.76%

Table 8. Experiment 2 – Inference Results

It was found as indicated from *Table 8* above that, hyperparameter optimisation of the XGB classification algorithm on the SGS dataset led to a reduction in the runtime by some 13.33% and shortening the entire inference process. Albeit, the reduction in test accuracy was a bit marginal as reported at 1.76%, the hyperparameter optimisation at the inference stage only saved a marginal energy of 3.59% which on the surface may seem to be very insignificant and therefore negligible. Other metrics measured, that is CPU duration and DRAM energy consumed at the inference stage had fair drops as recorded at 8.33% and 3.0% respectively, relative to experiment 1 where no hyperparameter optimisation was performed. The findings at the inference stage is inconsistent with that of Brownlee et al. [1] who found great energy savings of about 77% at the inference stage with one of the datasets used in their experimentation when hyperparameter optimisation was employed. This contradictory finding could possibly be explained by the differences in the ML algorithms used. Also, the different datasets among a host of other factors as used in this experiment and [1] could account for this inconsistency in findings.

4.3 Stand-alone Selected Features Optimised Model

As indicated in the research design, experiment 3 was conducted to answer research question 3 and to achieve objective 3. The results presented in this section is helpful in understanding how feature selection is able to make machine learning algorithms run faster while cutting down on the energy consumption usage in the process and in effect reducing the environmental cost with little or no harm to predictive accuracy. Recursive Feature Elimination with Cross Validation (RFECV) library provided by scikit-learn was used to carry out the feature optimisation in this specific experiment. All the features were re-checked to deal with the problem of dependences but none was found. Figure 7 below illustrates the performance of the RFECV in selecting the features and the number of features which were selected to be relevant in predicting the target.

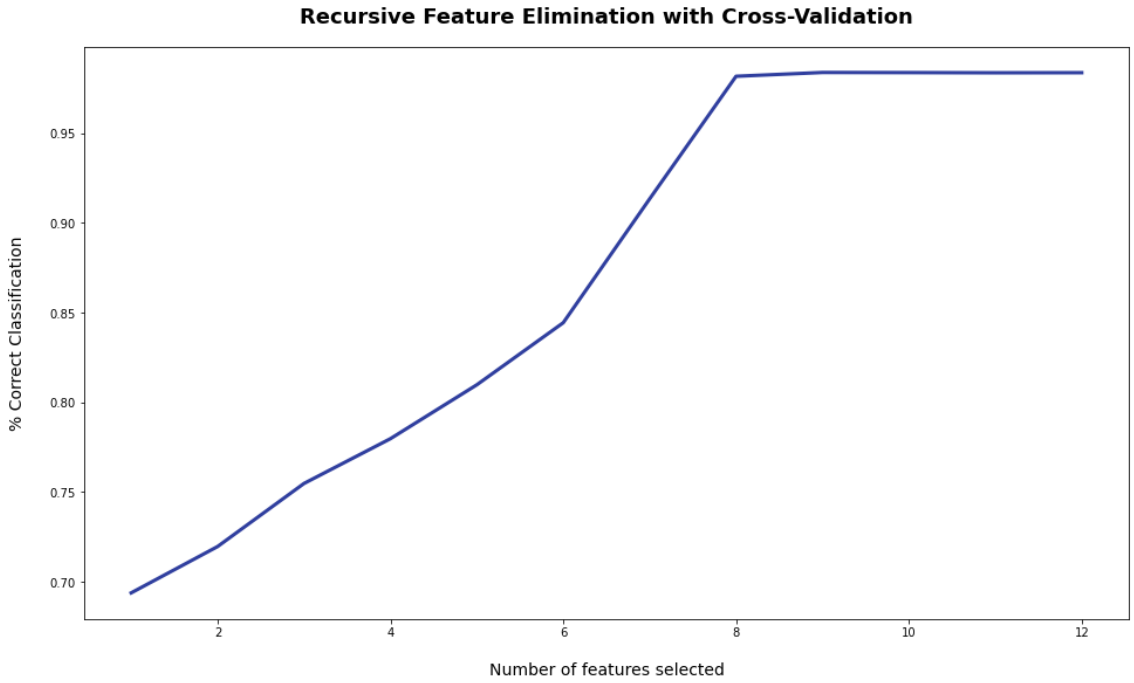

Figure 7. RFECV for Feature Optimisation

The above figure indicates that, accuracy did not improve after 9 features were used. What this result demonstrates is that, it is equally good to use nine features instead of the entire features of the SGS dataset and still be able to get same or even improved predictive model accuracy. As observed in Figure 8 below, features with indexes 5,6 and 7 representing the features *p2*, *p3* and *p4* were dropped as they did not contribute much in predicting the stability of a smart grid. From the description of the dataset in *Table 2*, it is not surprising that these features were dropped when RFECV was applied.

```
# Check which features were dropped ( Features that do not contribute Significantly in predicting the target)
print(np.where(rfecv.support_ == False)[0])

# Drop those features and get a new set of features for training
X.drop(X.columns[np.where(rfecv.support_ == False)[0]], axis=1, inplace=True)
```

```
[5 6 7]
```

Figure 8. Snippet of Code illustrating Features that were Eliminated

Following from the features that were eliminated by RFECV, it was appropriate that the relevance of the remaining features be known to fully appreciate RFECV's drop of features. The now reduced features is shown in Figure 9 below with each feature's relative importance in predicting the target, that is stability of smart grid as presented in the SGS dataset.

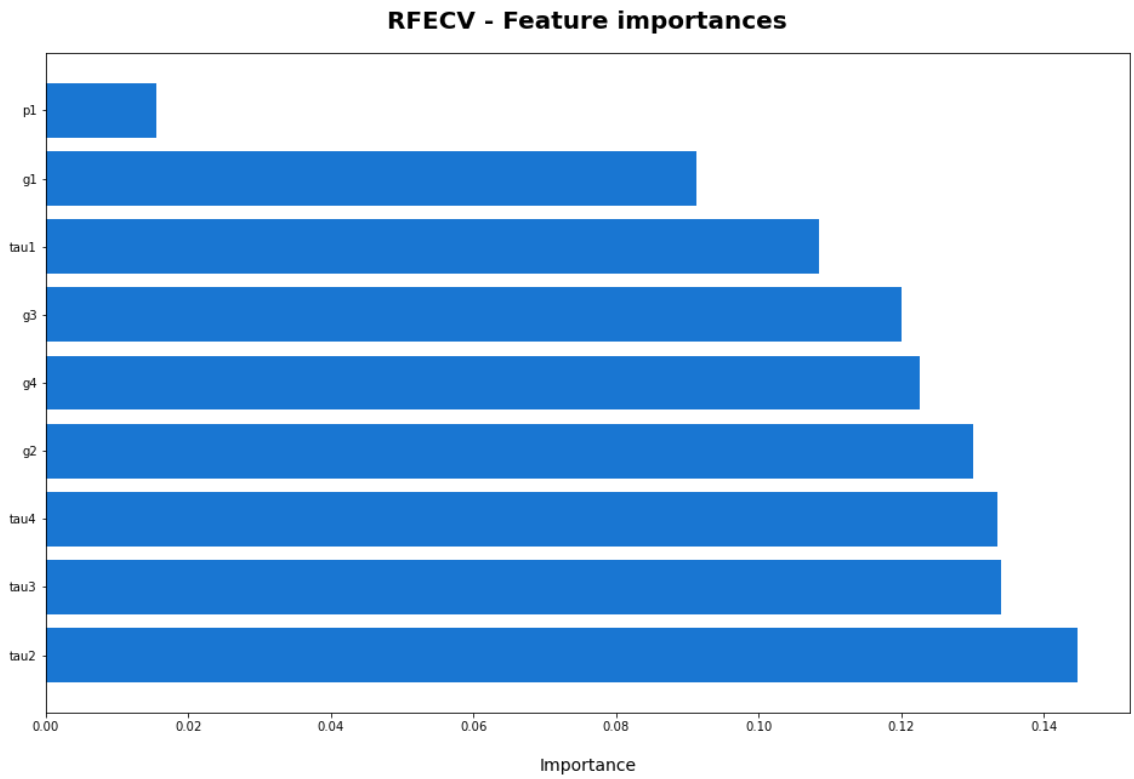

Figure 9. Relevance of the Selected Features

It is indicative from the *Figure 9* above that, the feature, *tau2* is the major contributor as shown by its importance in excess of 14% in predicting stability of smart grid. Also, the features *tau3*, and *tau4* and *g2* are seen to be strong contributors as in percentage-wise, they are nearing the importance of *tau2*. However, it is worthy of mentioning that with the exception of the features *p2* and *g1*, which recorded relatively low importance of approximately 2% and 9% respectively, all the other 7 features had importance in excess of 10% in predicting the smart grid stability target.

The next subsections present the standalone effects of the reduced features on the model runtime and the associated environmental cost at both ends of training and inference.

4.3.1 Training Results – Default hyperparameters on Selected Features of the Dataset

With the elimination of features that do not contribute significantly in predicting the target, the number of features employed and reported on in this section as results of experiment 3 is 9 instead of the original 12 features which were used in experiments 1 and 2. *Table 9* below shows the training results of experiment 3 which in part addresses research question 3 and goes to achieve objective 3 in part, which is to examine the extent by which feature optimisation on its own is useful in building faster and environmentally friendly AI models with little or no harm to the predictive power of the model that is being built.

Metric	Recorded	Relative Change
Training CPU Duration (S)	224.68	-10.92%
Training Runtime Duration (S)	10.68	-9.34%
Training CPU Energy Consumption (J)	1038.81	-11.27%
Training DRAM Energy Consumption (J)	52.65	-8.77%
Mean of 5-fold CV Accuracy (%)	97.77	-0.03%

Table 9. Experiment 3 – Training Results

The results found on experiment 3 and as presented in *Table 9* above shows that, using relevant subset of the entire features of the dataset could be helpful as a technique to cut down on the environmental cost associated with training AI models. From Table 9 above, after a reduction in number of features from 12 to 9, the time it took for the model to be fitted to the training dataset was shortened by 9.34%, saving CPU energy consumption of 11.27% and subsequently reducing the emission of GHG to the environment The shortening of the runtime and reduction in the energy consumption can be explained by the fact that an increased interpretability has been achieved with the subset of the entire features which makes it easy for the ML algorithm to learn with an improved learner performance. It is worthy to note that while these savings were made, the predictive accuracy was not adversely affected as a near negligible loss of only 0.03% in the predictive accuracy of the model that was constructed was found. Although, feature optimisation was applied in different fashions, the drop in energy consumption conforms to that which was found by Ghasemzadeh et al. [18]. Also, the other metrics had a fair reduction relative to experiment 1. CPU duration was made faster by 10.92% whereas the energy taken up by the DRAM was reduced by 8.77%. *Figure 10* below is used to highlight how feature optimisation performed on 30 repeated runs of fitting the model to the training dataset in experiment 3 relative to experiment 1 where the model was fitted to the entire features of the training dataset.

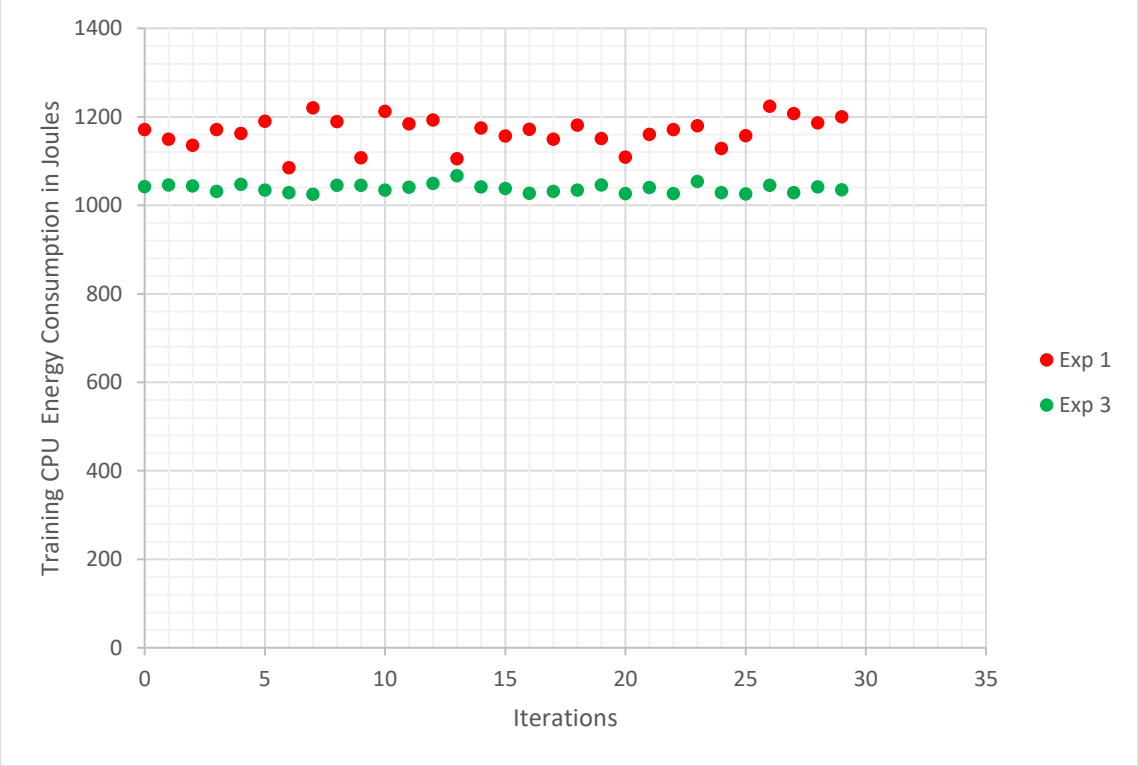

Figure 10. Training CPU Energy Consumption of Experiments 1 and 3 compared

It is shown in Figure 10 above that, albeit the gap between the energy taken up when the entire feature datasets are used as compared to that taken up when only features of importance are used is not that wide, however, none of the 30 repeated runs of fitting the model to the optimised features took up more energy in excess of that which were consumed in experiment 1 and accuracy also only suffered a near negligible loss. This is a strong indication of the need to apply feature optimisation technique as a candidate tool on the move towards faster and greener AI.

4.3.2 Inference Results – Default hyperparameters on Selected Features of the Dataset

To fully address research question 3 and complete achieve objective 3, this subsection augments subsection 4.3.1 to help shed more light on how feature optimisation impact on runtime and energy consumption when the model is being used to test on unseen dataset. *Table 10* below presents the findings of experiment 3 in regards to the inference stage of the machine learning process.

Metric	Recorded	Relative Change
Inference CPU Duration (S)	0.235	-2.08%
Inference Runtime Duration (S)	0.026	-12.97%
Inference CPU Energy Consumption (J)	1.648	-1.30%
Inference DRAM Energy Consumption (J)	0.098	-1.90%
Test Accuracy (%)	98.11	0.12%

Table 10. Experiment 3 – Inference Results

As presented in table 10 above, it was found that, on the inference side of the machine learning process, feature optimisation helped in lessening the runtime in testing the constructed model on unseen dataset by 12.97%. However, the drop in energy consumption as presented by the inference CPU energy consumption in the forementioned table was not that substantial as reported by pyRAPL as 1.3%. The other metrics also had a slight reduction as a 2.08% reduction was observed for the CPU duration and the energy consumption by the DRAM also was reduced by 1.90%. However, the predictive accuracy was found to have improved slightly by a margin of 0.12% relative to experiment 1 when all the features in the dataset were used. Though significant, this finding does not contribute much to confirming the use of feature optimisation alone as a technique for reducing the energy consumption at the inference side of the machine learning process.

4.4 Tuned Hyperparameters on Selected Features Optimised Model

This section presents the findings of the core objective of the dissertation. What this means is that, the findings on how the application of hyperparameter optimisation on selected features of the dataset help in reducing runtime and energy consumption which by and large, reduce the environmental cost in the quest of building highly accurate models is presented in this section. The *GridSearchCV* as provided by scikit-learn and the search space as used in experiment 1, as seen in *Table 1*, were also used for the hyperparameter optimisation in experiment 4 which addresses research question 4. Further to this, the 9 selected features of the dataset in experiment 3, were repeated for experiment 4 instead of the entire 12 features of the original dataset. As seen in *Table 11* below, the values returned as best estimators after the exhaustive search using the *GridSearchCV* on the 9 selected features were same as that returned for the entire 12 features.

XGB Hyperparameter	Best Configuration Value
Colsample_bytree	0.5
Learning_rate	0.2
Max_depth	5
Min_child_weight	2
N_estimators	100
subsample	1

Table 11. Experiment 4 – Best Hyperparameter Configuration

Following from the best hyperparameter configuration as presented in *Table 11* above, the findings in respect of hyperparameter optimisation on selected features' impact on runtime and energy consumption at both extremes of training and inference is presented in subsections 4.4.1 and 4.4.2 respectively

4.4.1 Training Results – Tuned hyperparameters on Selected Features of the Dataset

In an attempt to examine the combined effect of the application of hyperparameter optimisation and feature optimisation as possible tools to employ in building accurate, faster and energy efficient AI models, the training results presented in *Table 12* below shows how a blend of the two techniques performed in achieving the overall aim when XGB classification model was fitted to the training data of the selected features of the SGS dataset.

Metric	Recorded	Relative Change
Training CPU Duration (S)	171.27	-32.09%
Training Runtime Duration (S)	8.14	-30.88%
Training CPU Energy Consumption (J)	760.16	-35.07%
Training DRAM Energy Consumption (J)	39.44	-31.67%
Mean of 5-fold CV Accuracy (%)	95.91	-1.93%

Table 12. Experiment 4 – Training Results

As presented in *Table 12* above, it was found that combining hyperparameter optimisation and feature optimisation to fit the model to the training dataset, runtime was reduced by 30.88%. Furthermore, the energy used in fitting the model to the training dataset as measured by pyRAPL was also cut down by 35.07%. With these impressive reduction in the training runtime and energy consumption which in effect lead to a substantial reduction in the environmental cost, a marginal loss of 1.93% in predictive accuracy was recorded. This impressive savings in the runtime and energy consumption which in effect cut down on the overall environmental cost could be explained by the XGB algorithm's ability to complete the learning process faster due to the higher explainability achieved through the blend of the 2 techniques. It is worth mentioning also that, the CPU duration in the process was shortened by 32.09% whereas the energy usage by the DRAM as reported by pyRAPL also made a significant savings of 31.67%. *Figure 11* below, illustrates the energy consumption of the 30 repeated runs performed in fitting the tuned hyperparameters model on the selected features as compared to experiment 1, where default

hyperparameters of the XGB classification algorithm were used on the entire features of the dataset.

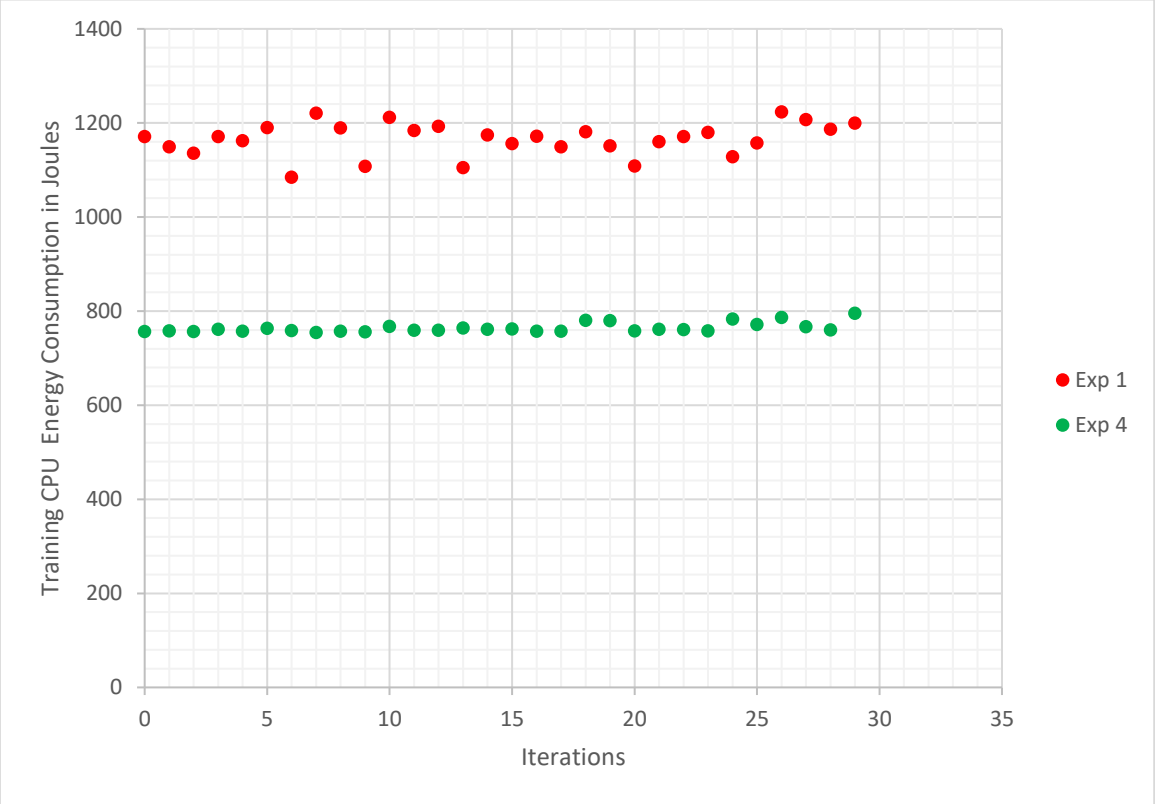

Figure 11. Training CPU Energy Consumption of Experiments 1 and 4 compared

In comparing the results of experiment 4 against the benchmark model which is experiment 1 after the 30 repeated fitting of the model to the training dataset, it was revealed as illustrated in Figure 11 above that, none of the 30 drops in the energy consumption happened by chance when hyperparameter optimisation was applied on the selected features. This makes combination of hyperparameter optimisation and feature optimisation strong candidate tools for use in achieving accurate, faster and eco-friendly AI.

4.4.2 Inference Results – Tuned hyperparameters on Selected Features of the Dataset

To be able to appreciate fully the effect of hyperparameter optimisation and feature optimisation on the other side of the ML process which is inference energy consumption and runtime, the findings of experiment 4 relating to testing the model constructed from the tuned hyperparameters on the selected features on unseen data is presented in *Table 13* below.

Metric	Recorded	Relative Change
Inference CPU Duration (S)	0.22	-7.65%
Inference Runtime Duration (S)	0.025	-15.38%
Inference CPU Energy Consumption (J)	1.61	-3.80%
Inference DRAM Energy Consumption (J)	0.097	-3.39%
Test Accuracy (%)	95.84	-2.19%

Table 13.. Experiment 4 – Inference Results

From *Table 13* above, it is shown that after tuning the hyperparameters on the selected features and using the trained model to test on unseen and untouched model, the runtime was shortened by 15.38% as compared to when the default model was used on the entire features of the dataset. More so, the energy taken up by the CPU which is indicative of the environmental cost had a small savings of 3.39%. The other inference metrics measured also made fair savings as observed in table 13 above where CPU duration dropped by 7.65% and the energy consumed by the DRAM also dropped by 3.39%. However, the inference accuracy was dropped relative to when no hyperparameter configuration was made and the entire features of the dataset were used in developing the model by 2.19%. Again, this finding does not seem to favour the generalisation of a blend of hyperparameter optimisation and feature optimisation as strong techniques to employ in when matters of building testing ML models on unseen datasets with minimal adverse effects on planet earth as the focus as though completion time dropped, energy consumption did not change that much and predictive accuracy was a bit distracted.

4.5 General Results and Discussion

This subsection of the dissertation focuses on the overall results following from the objective-based results above presented. Also, a thorough discussion of the findings of the dissertation is made in this particular subsection as well. As the broader focus is to make AI faster and greener while trying to maintain predictive accuracy, hyperparameter optimisation technique as used by researchers such as Brownlee et al. [1] and feature optimisation technique as used by researchers such as Ghasemzadeh et al. [18] were considered in series of four experimentations to explore how they can be used to achieve the broader aim. In this dissertation, the runtime as proxied by the wall-clock equivalent time in seconds and the energy consumption is proxied by the CPU energy consumption as measured in microjoules by pyRAPL and subsequently converted to Joules as well as the predictive accuracy will be the main focus of discussion in this section, although, other metrics such as CPU Duration, DRAM energy consumption were also measured.

Experiment 1 was setup to address objective 1. What this means is that, the default hyperparameters of the XGB classification algorithm was used on the entire features of the SGS dataset to explore the actual runtime and the environmental cost involved during the training and inference stages of the machine learning process. Experiment 2 was setup to address objective 2. In this case, hyperparameter optimisation using GridSearchCV as provided by scikit-learn was explored on the entire features of the SGS dataset to examine how it can be used to cut down the energy consumed and the runtime at both extremes of training and inference with no or minimal harm to the predictive accuracy of the model being constructed. More so, experiment 3 was setup to address objective 3. In this setup, feature optimisation using RFECV as provided by scikit-learn was used on the SGS dataset to cut down the number of features used in building the model. The XGB hyperparameters were kept at defaults and then the runtime and energy consumption were measured at both ends of training and inference to investigate how feature optimisation can be helpful as a technique to employ towards a faster and greener AI. Finally,

experiment 4 was setup to address objective 4. In this setup, the combined effects of hyperparameter optimisation and feature optimisation were examined on runtime and energy consumption to appreciate how combining these 2 techniques could be helpful in achieving the broader aim. Tables 14 and 15 below represents a snapshot of the findings of the series of experimentations at both ends of training and inference respectively.

Metric	**Exp. 1**	**Exp. 2**	**Exp. 3**	**Exp. 4**
Training CPU Duration (S)	252.21	179.29	224.68	171.27
Training Runtime Duration (S)	11.78	8.67	10.68	8.14
Training CPU Energy Consumption (J)	1170.73	819.68	1038.81	760.16
Training DRAM Energy Consumption (J)	57.71	42.27	52.65	39.44
Mean of 5-fold CV Accuracy (%)	97.80	96.10	97.77	95.91

Table 14.. Experiment 1 to 4 – Training Results

Metric	**Exp. 1**	**Exp. 2**	**Exp. 3**	**Exp. 4**
Inference CPU Duration (S)	0.240	0.220	0.235	0.222
Inference Runtime Duration (S)	0.030	0.026	0.026	0.025
Inference CPU Energy Consumption (J)	1.670	1.610	1.648	1.607
Inference DRAM Energy Consumption (J)	0.100	0.097	0.098	0.097
Test Accuracy (%)	97.990	96.270	98.110	95.842

Table 15. Experiment 1 to 4 – Inference Results

It can be observed from Tables 14 and 15 above that the standalone effect of hyperparameter optimisation on runtime and energy consumption is substantial. This conforms to findings in other works such as the works by [1], [13]. Also, from the same tables, it can be noticed that, the standalone impact of feature optimisation although, improved predictive accuracy but was not much effective in reducing runtime and energy consumption at both ends of training and inference. The combination of the two techniques as noted in experiment 4 results for both training and inference is of great importance on the move towards energy efficient, faster and also accurate models with minimal harm to planet earth. On examination of the runtime of the two extremes in the ML process, it was revealed that it required *393 times* the duration it takes for a model to be tested on unseen data for a model to be fitted to the training data. Also, on exploring the energy consumption at both extremes, it was observed that, *701 times* of the energy consumed to test on unseen data was required to fit the model to the training dataset. The longer duration and relatively higher energy consumptions found at the training stage as compared to the testing stage is consistent with machine learning expectations. This could be explained by the processes that the algorithm has to go through to learn the features of the training dataset and possibly the size of the training dataset could account for the longer period and huge energy consumption. These findings indicate that training a model is very costly both computationally and environmentally as compared to testing the constructed model on the unseen dataset. This finding conforms to that of Brownlee et al [1]. To this end, it is essential that much focus is

channelled to reducing the computational and energy cost and for that matter the environmental cost attributable to model training than testing.

Looking at the results in Tables 14 and 15 alone, it can be fully confirmed that hyperparameter optimisation and feature optimisation are vital tools to employ in building machine learning models that are not only accurate but also faster and greener. However, noise from several environmental factors can have great influence on a running system which in a way could be problematic to the validity of the research carried out. This necessitated the need for the 30 and 1000 repeated runs for training and inference phases of the ML process respectively in order to be very certain of the findings of this research. For a thorough discussion therefore, the need to use statistical approaches in evaluating the results obtained was invaluable. Going forward, much emphasis is placed on runtime and the CPU energy consumption for both training and inference. However, greater portion of the discussion will be based on the training runtime and energy consumption since the ML training phase is the part that takes longer to complete and also very costly environmentally as proxied by its energy usage and found in this dissertation.

Tables 16 and 17 indicate the results of the *Mann-Whitney U test* performed to check for statistical significance of the reductions observed as reported above for runtime and energy consumption at both ends of training and inference respectively in order to be sure that, they were not recorded by chance.

Training Experiments	**Metric**	**Mann-Whitney U Test**		**Remarks**
		Hypothesis	**Results (ρ-value)**	
Exp 1 *vs* Exp 2	Runtime	H_0: From Same Distribution	0.00	Reject H_0
	Energy Consumption	H_A: From Different Distribution	0.00	Reject H_0
Exp 1 *vs* Exp 3	Runtime	H_0: From Same Distribution	0.00	Reject H_0
	Energy Consumption	H_A: From Different Distribution	0.00	Reject H_0
Exp 1 *vs* Exp 4	Runtime	H_0: From Same Distribution	0.00	Reject H_0
	Energy Consumption	H_A: From Different Distribution	0.00	Reject H_0

Table 16. Mann-Whitney U Test of Significance – Training Results

Inference Experiments	Metric	Mann-Whitney U Test		Remarks
		Hypothesis	Results (ρ-value)	
Exp 1 *vs* Exp 2	Runtime	H_0: From Same Distribution	0.00	Reject H_0
	Energy Consumption	H_A: From Different Distribution	0.00	Reject H_0
Exp 1 *vs* Exp 3	Runtime	H_0: From Same Distribution	0.00	Reject H_0
	Energy Consumption	H_A: From Different Distribution	0.00	Reject H_0
Exp 1 *vs* Exp 4	Runtime	H_0: From Same Distribution	0.00	Reject H_0
	Energy Consumption	H_A: From Different Distribution	0.00	Reject H_0

Table 17. Mann-Whitney U Test of Significance – Inference Results

From the Mann-Whitney U test conducted the results of which are as shown in Tables 16 and 17 for both training and inference respectively, there is sufficient evidence to reject the null hypothesis (H_0) in favour of the alternate hypothesis (H_A) in all of the experiments compared. What this means is that, the differences in the medians of the runtime between experiments 1 and 2 is significant at the training phase of the experimentation. Also, the difference in the medians of the training CPU energy consumptions of experiments 1 and 2 is significant and did not happen by chance. This interpretation also applies to the inference stage as well. Again, the interpretation applies to the comparison of experiments 1 versus 3 and experiments 1 versus 4 at both ends of training and inference. The above results confirm that hyperparameter optimisation and feature optimisation are good candidate techniques to consider when building models with higher accuracy that takes into account speed and the need to reduce the emission of GHG and its devastating effects on the planet earth.

The violin plots illustrated in *Figures 12 - 15* below are visual representations of the energy consumptions for the 30 and 1000 repeated runs at both ends of training and inference. Whereas, Figures 12 and 13 are for the energy consumptions at both extremes of training and inference respectively, Figures 14 and 15 are for the runtimes at both ends of training and inference respectively as well. The violin plots are indicative of the relationship of each of the four experiments to energy consumption and runtime at both ends of training and inference as earlier mentioned.

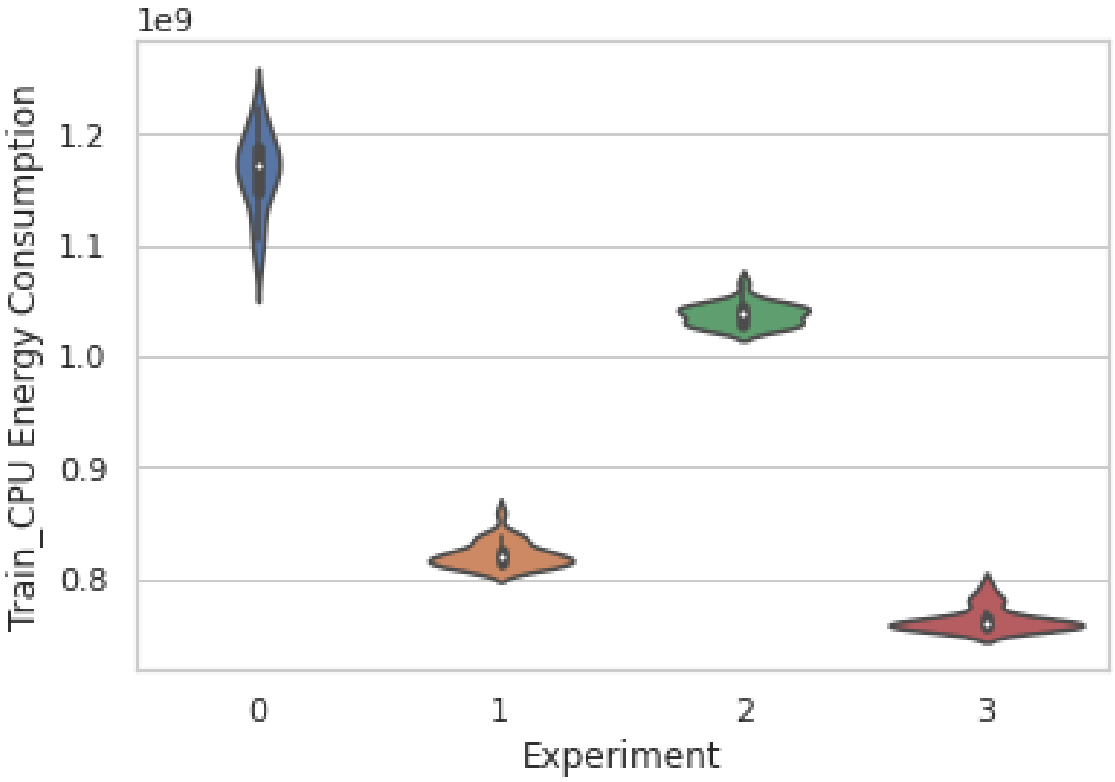

Figure 12. Distribution of Experiments 1 – 4 Training Energy Consumption

It is clear from the violin plot in *Figure 12* that the median training energy consumption for experiment 4 (see index 3 on the experiment axis) is lower than the other three experiments conducted. The experiment 4 is indicative of the effects of hyperparameter optimisation on selected features. From the same figure 12, it is also clear that the median training energy consumption for experiment 1 as noted with index 0 on the experiment axis is the highest. Experiment 1 is the case of default hyperparameters on the entire features of the dataset used. In the case of the shape of the distributions, it can be observed from Figure 12 above that the training energy consumptions of each of the 4 experiments are highly concentrated around the median as seen with the wide distributions in the middle of the violins in each of the experiments. The observations in Figure 12 therefore add credence to the fact that the distributions of the experiments compared are not equal. Similar interpretation applies to Figures 13, 14 and 15 below.

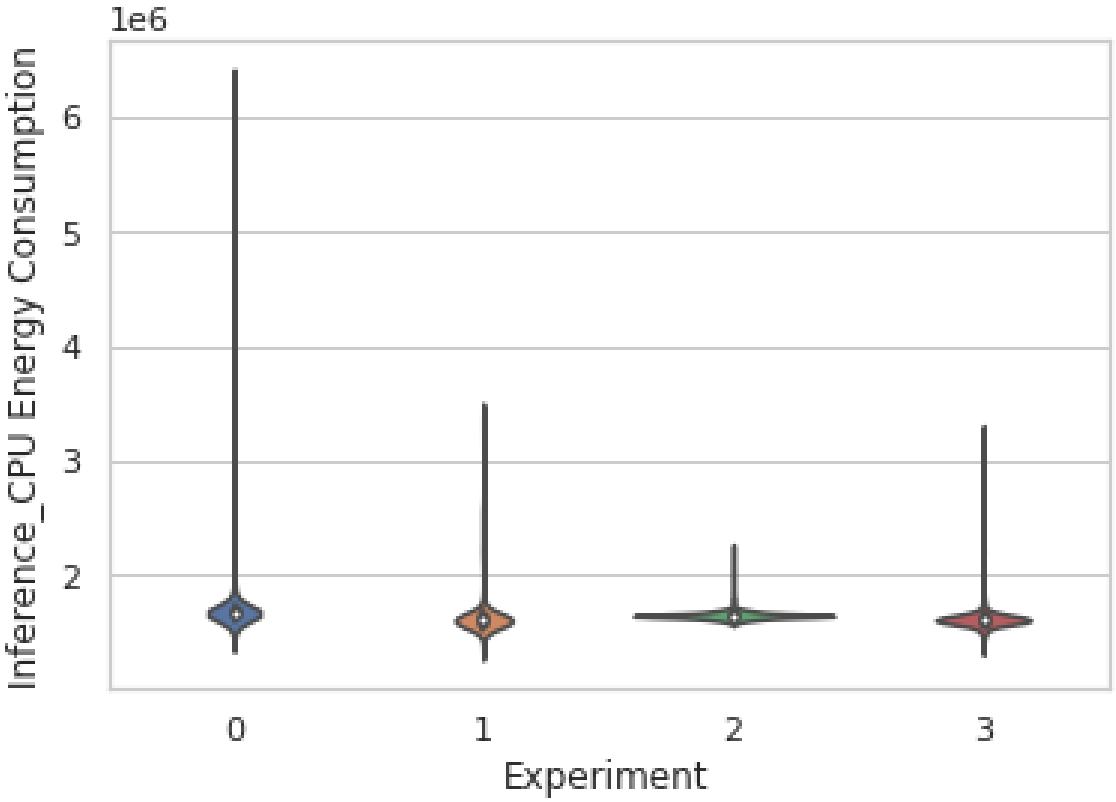

Figure 13. **Distributions of Experiments 1 – 4 Inference Energy Consumption**

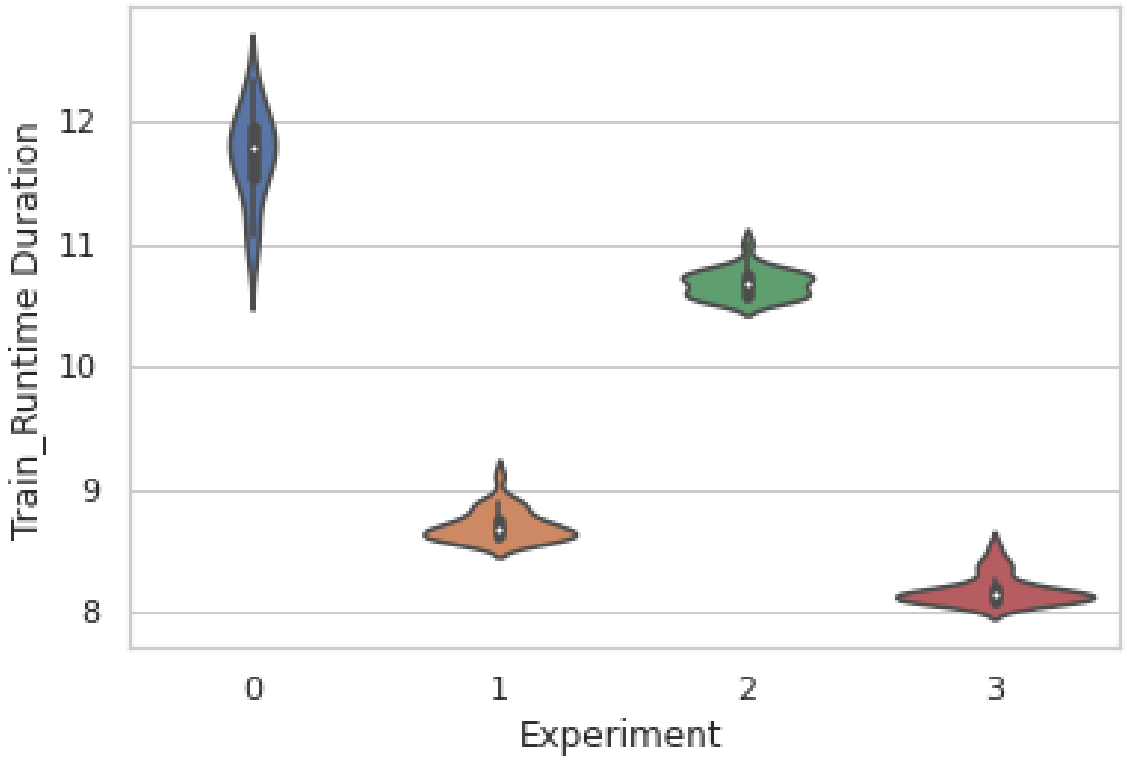

Figure 14. **Distribution of Experiments 1 – 4 Training Runtime**

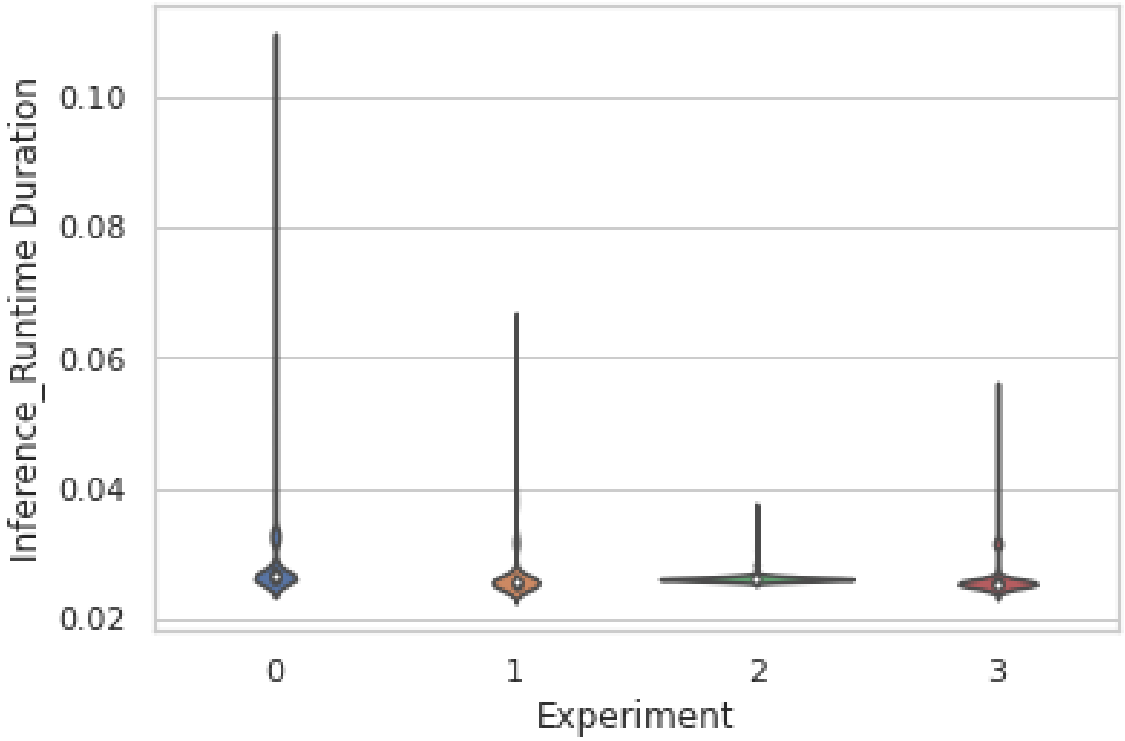

Figure 15. Distribution of Experiments 1 – 4 Inference Runtime

Again, *Figure 16* below sheds more light on the impact of hyperparameter optimisation and feature optimisation techniques to cutting down the energy consumption while keeping the predictive accuracy. It is an overall pictorial view of the 4 experiments performed.

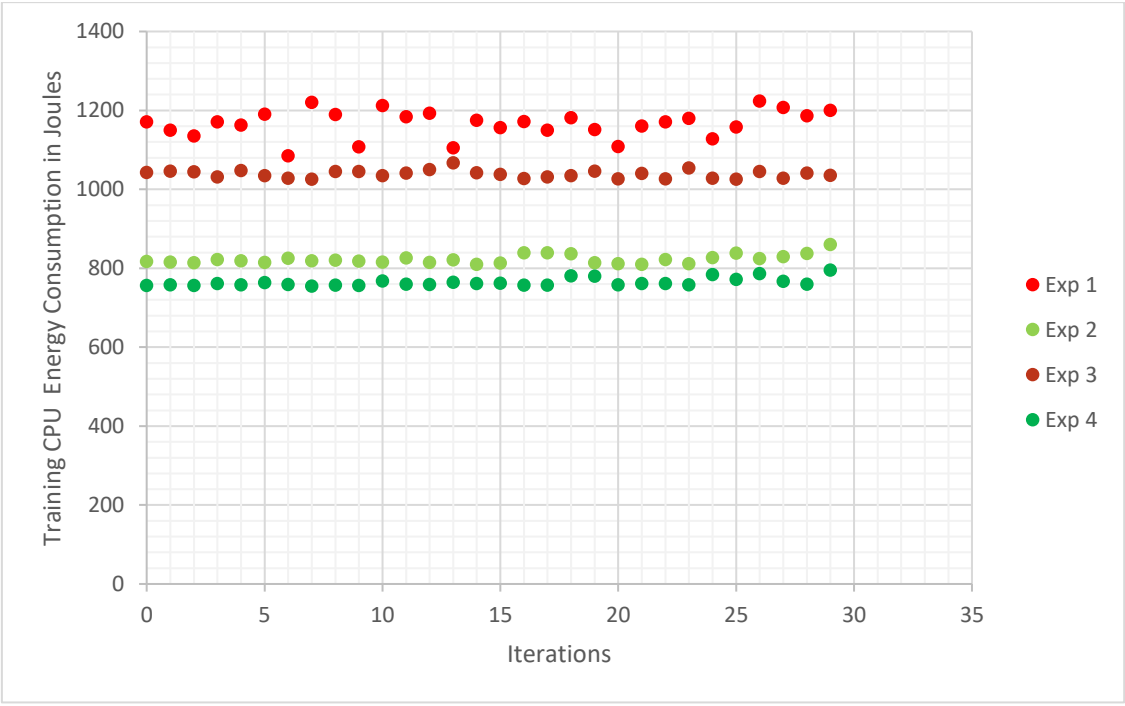

Figure 16. Training Energy Consumption Visualised (Experiment 1 – 4)

From the visual representation in Figure 16 above, it can be observed that combining hyperparameter optimisation and feature optimisation was able to reduce the energy consumed and its subsequent effect of emissions of GHGs to the environment better than the two individual techniques on their own as earlier explained in this dissertation.

5 Conclusion

This section is the concluding section of the dissertation. It presents the summary of the findings, evaluation of the achievements and recommendations for future studies.

5.1 Summary

This dissertation aimed at approaches to faster and greener AI with no or minimal loss in predictive accuracy. Hyperparameter optimisation and feature optimisation were explored as the techniques to achieving this aim. The XGB classification algorithm was used on the SGS dataset in four different experimental series.

In experiment 1, it was found that it takes longer for a machine learning algorithm to be fitted to the training data than it takes for it to be tested on unseen data by 393times. Again experiment 1 also indicated that the energy consumed in fitting a model to a dataset takes as much as 701times the energy it takes the model to be tested on unseen data. Experiment 1 has helped to unravel which aspect of the machine learning process consumes more energy and subsequently contribute to emitting large quantities of GHG which has adverse effects on our planet.

More so, experiment 2 also has shown that hyperparameter optimisation alone has the capacity to cut down on the runtime and energy consumption during the training phase of the machine learning process. In experiment 2, it was shown that hyperparameter optimisation can shorten the time it takes to fit a model to the training data by 26.4% whereas energy of 29.99% was saved in fitting the model to the training data with a minimal reduction of 1.74% in predictive accuracy. This experiment has shown that hyperparameter optimisation is an effective approach to employ on the move towards faster and environmentally friendly AI with just a little loss in predictive accuracy.

Also, it was found from experiment 3 that, using relevant subset of the entire features of the dataset to fit the model to the training data by using feature optimisation can contribute slightly to reducing the runtime and energy consumption by 10.92% and 11.27% respectively with a near negligible loss in predictive accuracy from 97.8% to 97.77%. This experiment has demonstrated that feature optimisation can be a useful approach to apply in building AI models that compute faster and with an overall reduced harm to the environment.

Again, it was found through experiment 4 that, combining hyperparameter optimisation and feature selection could make models run faster and with much more reduced harm to the environment than the individual techniques on their own (see Figure 16). It was revealed in experiment 4 that, performing hyperparameter optimisation on the selected features made it possible for the model to be fitted to the training data with a shortened time of 30.88%. Furthermore, savings of 35.07% was made in the energy used in fitting the model to the training dataset which in effect leads to a reduction in the emission of GHG to planet earth in an approximated equal measure. It is worthy of mentioning that the predictive accuracy at the point only dropped marginally from 97.80% to 95.91%

Finally, it was revealed through the 4 experimental series that, little could be achieved to make the model run faster and with much higher reduced energy consumption when the model is being used to test on unseen and untouched data although in all of the experiments, marginal reductions were recorded for both runtime and energy consumption except for experiment 4, that is the application of hyperparameter optimisation on the selected features, which had a significant reduction in the runtime by some 15.38% with energy savings made as only 3.39% whereas predictive accuracy suffered a loss by a margin of 2.19%.

5.2 Evaluation

The broader aim of making AI faster and greener was approached on these original objectives which were set as follows:

1. To investigate runtime and energy consumption of XGB classifier algorithm on the SGS dataset during training and inference.
2. To explore the impact of hyperparameter optimisation on the runtime and energy consumption of XGB classifier algorithm on the SGS dataset during training and inference.
3. To examine the effects of feature optimisation on runtime and energy consumption of XGB classifier algorithm on the SGS dataset during training and inference.
4. To explore the combined effect of hyperparameter optimisation and feature optimisation on the runtime and energy consumption of XGB classifier algorithm on SGS dataset at both extremes of training and inference.

Objective 1 was met through experiment 1. The runtime and energy consumption of the XGB classification algorithm which was used to train and test on the SGS dataset were recorded with the *time.time()* library and *pyRAPL* software tool kit respectively. The investigation led to the revelation that, training a model takes much time and extremely higher energy to complete than testing the model on unseen data. What this means is that, the environmental cost associated with training a model is excessively huge and therefore calls for much attention than testing the model on unseen data.

Also, objective 2 was well achieved through experiment 2. Hyperparameter optimisation was applied on the entire features of the dataset and the runtime and energy consumption were measured. It was observed that hyperparameter on its own was capable of cutting down drastically the runtime and energy taken up to fit the XGB classification algorithm to the entire features of the dataset while trading-off a little of predictive accuracy. However, the reduction for runtime and energy consumption on inference was minimal. This is an indication that, hyperparameter optimisation is a strong candidate technique to apply towards faster and greener AI.

Again, objective 3 was also met through experiment 3. Although, the impact of feature optimisation as a technique for the reduction in runtime and energy consumption was achieved at both extremes of fitting the model to the training data and testing the model on unseen data, the effects were not that much as compared to that which was achieved by the hyperparameter optimisation technique. It can be said however that, accuracy was almost maintained when feature optimisation was employed on the XGB classification algorithm on the SGS dataset. This demonstrates feature optimisation as a technique to apply in machine learning for developing accurate, faster and eco-friendly models although its impact is not that substantial relative to hyperparameter optimisation technique.

Finally, objective 4 was also met through experiment 4. It was revealed that, the combination of hyperparameter optimisation and feature optimisation could have significant impact on runtime and energy consumption at both extremes of training and inference. Most especially the reduction in runtime and energy consumption as recorded when fitting the XGB classification algorithm to the training data was very substantial than the individual techniques on their own. This has shown that, combining hyperparameter optimisation and feature optimisation techniques could help make models equally accurate, faster and with overall reduced harm to our planet especially when it comes to the training phase where much emissions of GHGs take place.

5.3 Future Work

This dissertation seems to be one of the first research works geared towards approaches to minimising the environmental cost associated with faster but also accurate AI. Although pyRAPL has

been used to conveniently measure the energy taken up by the CPU to run at both ends of training and inference, leading to excellent results which were used as proxy for the environmental cost, the direct quantity of GHGs emitted, indicating the actual environmental cost could not be measured reliably. Although some researchers have formulated procedures and other technologies for this measure, these has not been generalised yet. This situation could be improved if researchers would take delight in coming out with technologies that directly quantify the environmental cost instead of using CPU time as done in other works and CPU energy consumption as used in this research as proxies.

Although, impressive results were achieved to help in acknowledging the use of hyperparameter optimisation and feature optimisation as techniques to employ towards faster and greener AI with no or minimal harm to predictive accuracy, only one specific dataset, was used to carry out the experiment due to time constraints. To be able to generalise this dissertation further, the approach could be employed on other large and especially energy hungry datasets as well.

Another threat to validity of this research is the specific system hardware used and the location of the system hardware. Other system hardware components may behave differently leading to different levels of energy usage. Also, different countries may have different sources and proportions of energy generation other than the UK which in effect leads to differences in contributions to the harm caused to planet earth by different countries. Varying this research using different system hardware based in a different country would help in generalising the findings of this research.

The XGB classification algorithm was the only machine learning algorithm used in this experimentation. Albeit, it is very popular in its usage among players in the AI community for various projects and gave excellent results in this dissertation as its hyperparameters could be tuned, exploring other machine learning algorithms such as logistic regression, random forest and the likes with the approaches in this research however, will help in establishing a formidable knowledge on the use of hyperparameter optimisation and feature optimisation as techniques to employ to build accurate, faster and environmentally-friendly AI.

More so, the optimisation techniques and python libraries used for the hyperparameter configuration and feature selection have helped in achieving the objectives of this research. The limitation here however, has to do with the finite search space and the exhaustive search approach used by the GridSearchCV optimisation library for the hyperparameter tuning. It will be interesting to explore how other search optimisation techniques such as Bayesian search, random search and even other genetic algorithms help in achieving the objectives and the overall aim of making AI faster and greener. In same vein, exploring other optimisation techniques and python libraries for the feature selection other than the RFECV provided by scikit-learn as used in this research will go a long way to generalise the findings of this dissertation.

Finally, making explainability of the key drivers of runtime and energy consumption in building AI models an added objective to this dissertation, would help all players and even users of AI to understand, interpret and find best approaches to cutting down on such energy hotspots so as to make the dream of faster and greener AI a reality while maintaining predictive accuracy.

References

[1] A. E. . Brownlee, J. Adair, S. O. Haraldsson, and J. Jabbo, “Exploring the Accuracy – Energy Trade-off in Machine Learning,” pp. 11–18, 2021, doi: 10.1109/gi52543.2021.00011.

[2] W. Ertel, “Introduction to Artificial Intelligence,” 2017, doi: 10.1007/978-3-319-58487-4.

[3] S. Das, A. Dey, and N. Roy, “Applications of Artificial Intelligence in Machine Learning: Review and Prospect,” *Int. J. Comput. Appl.*, vol. 115, no. 9, pp. 975–8887, 2015.

[4] A. Alimadadi, S. Aryal, I. Manandhar, P. B. Munroe, B. Joe, and X. Cheng, “Artificial intelligence and machine learning to fight covid-19,” *Physiol. Genomics*, vol. 52, no. 4, pp. 200–202, 2020, doi: 10.1152/physiolgenomics.00029.2020.

[5] Y. Wu, X. Xue, L. Le, X. Ai, and J. Fang, “Real-time Energy Management of Large-scale Data Centers: A Model Predictive Control Approach,” in *2020 IEEE Sustainable Power and Energy Conference (iSPEC)*, 2020, pp. 2695–2701, doi: 10.1109/iSPEC50848.2020.9351010.

[6] M. Zakarya, I. U. Rahman, and A. A. Khan, “Energy crisis, global warming amp; IT industry: Can the IT professionals make it better some day? A review,” in *2012 International Conference on Emerging Technologies*, 2012, pp. 1–6, doi: 10.1109/ICET.2012.6375501.

[7] G. A. Florides and P. Christodoulides, “Global warming and carbon dioxide through sciences,” *Environ. Int.*, vol. 35, no. 2, pp. 390–401, 2009, doi: https://doi.org/10.1016/j.envint.2008.07.007.

[8] IPCC, “IPCC report Global warming of 1.5°C,” *Ipcc - Sr15*, vol. 2, no. October, pp. 17–20, 2018, [Online]. Available: www.environmentalgraphiti.org.

[9] “Climate Change Act 2008.”

[10] “The Climate Change Act 2008 (2050 Target Amendment) Order 2019.”

[11] E. Strubell, A. Ganesh, and A. McCallum, “Energy and policy considerations for modern deep learning research,” *AAAI 2020 - 34th AAAI Conf. Artif. Intell.*, no. 1, pp. 1393–13696, 2020, doi: 10.1609/aaai.v34i09.7123.

[12] P. K. D. Pramanik *et al.*, “Power Consumption Analysis, Measurement, Management, and Issues: A State-of-the-Art Review of Smartphone Battery and Energy Usage,” *IEEE Access*, vol. 7, pp. 182113–182172, 2019, doi: 10.1109/ACCESS.2019.2958684.

[13] A. E. I. Brownlee, N. Burles, and J. Swan, “Search-Based Energy Optimization of Some Ubiquitous Algorithms,” *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 1, no. 3, pp. 188–201, 2017, doi: 10.1109/TETCI.2017.2699193.

[14] X. Dai *et al.*, “ChamNet: Towards efficient network design through platform-aware model adaptation,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2019-June, pp. 11390–11399, 2019, doi: 10.1109/CVPR.2019.01166.

[15] E. G. Martín, *Energy Efficiency in Machine Learning Approaches to Sustainable Data Stream Mining*. 2019.

[16] D. Stamoulis, E. Cai, D. Juan, D. Marculescu, and M. View, “HyperPower : Power- and Memory-Constrained Hyper-Parameter Optimization for Neural Networks,” pp. 19–24, 2018.

[17] S. K. Saha, S. Sarkar, and P. Mitra, “Feature selection techniques for maximum entropy

based biomedical named entity recognition," *J. Biomed. Inform.*, vol. 42, no. 5, pp. 905–911, 2009, doi: 10.1016/j.jbi.2008.12.012.

[18] H. Ghasemzadeh, N. Amini, R. Saeedi, and M. Sarrafzadeh, "Power-Aware Computing in Wearable Sensor Networks: An Optimal Feature Selection," *IEEE Trans. Mob. Comput.*, vol. 14, no. 4, pp. 800–812, 2015, doi: 10.1109/TMC.2014.2331969.

[19] "Smart Grid Stability | Kaggle." https://www.kaggle.com/pcbreviglieri/smart-grid-stability (accessed Sep. 10, 2021).

[20] N. J. Nilsson, *Principles of Artificial Intelligence*, vol. PAMI-3, no. 1. 1981.

[21] M. Learning and D. Mining, "Machine Learning and Data Mining 8," vol. 12, no. Comp 5318, pp. 3–12, 2017.

[22] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science (80-.).*, vol. 349, no. 6245, pp. 255–260, Jul. 2015, doi: 10.1126/SCIENCE.AAA8415.

[23] "Climate Change | National Geographic Society." https://www.nationalgeographic.org/encyclopedia/climate-change/ (accessed Sep. 05, 2021).

[24] "Effects | Facts – Climate Change: Vital Signs of the Planet." https://climate.nasa.gov/effects/ (accessed Sep. 05, 2021).

[25] J. Delbeke, A. Runge-Metzger, Y. Slingenberg, and J. Werksman, "The paris agreement," *Towar. a Clim. Eur. Curbing Trend*, pp. 24–45, 2019, doi: 10.4324/9789276082569-2.

[26] E. Cai, D. C. Juan, D. Stamoulis, and D. Marculescu, "NeuralPower: Predict and deploy energy-efficient convolutional neural networks," *J. Mach. Learn. Res.*, vol. 77, pp. 622–637, 2017.

[27] C. Zhang, A. Hindle, and D. M. German, "The impact of user choice on energy consumption," *IEEE Softw.*, vol. 31, no. 3, pp. 69–75, 2014, doi: 10.1109/MS.2014.27.

[28] A. Carroll and G. Heiser, "An Analysis of Power Consumption in a Smartphone Motivation," 2010.

[29] A. Carroll and G. Heiser, "The systems hacker's guide to the Galaxy energy usage in a modern smartphone," *Proc. 4th Asia-Pacific Work. Syst. APSys 2013*, 2013, doi: 10.1145/2500727.2500734.

[30] M. Anderson and C. Gómez-Rodríguez, "Distilling Neural Networks for Greener and Faster Dependency Parsing," pp. 2–13, 2020, doi: 10.18653/v1/2020.iwpt-1.2.

[31] M. A. Bokhari, B. R. Bruce, B. Alexander, and M. Wagner, "Deep parameter optimisation on android smartphones for energy minimisation-a tale ofwoe and a proof-of-concept," *GECCO 2017 - Proc. Genet. Evol. Comput. Conf. Companion*, pp. 1501–1508, 2017, doi: 10.1145/3067695.3082519.

[32] A. Candelieri, R. Perego, and F. Archetti, "Green machine learning via augmented Gaussian processes and multi-information source optimization," *Soft Comput.*, vol. 7, 2021, doi: 10.1007/s00500-021-05684-7.

[33] Y. H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks," *IEEE J. Solid-State Circuits*, vol. 52, no. 1, pp. 127–138, 2017, doi: 10.1109/JSSC.2016.2616357.

[34] G. Ding, J. Tian, J. Wu, Q. Zhao, and L. Xie, "Energy efficient human activity recognition using wearable sensors," *2018 IEEE Wirel. Commun. Netw. Conf. Work. WCNCW 2018*,

pp. 379–383, 2018, doi: 10.1109/WCNCW.2018.8368980.

[35] Y. H. Lu, A. C. Berg, and Y. Chen, “Low-Power image recognition challenge,” *AI Mag.*, vol. 39, no. 2, pp. 87–88, 2018, doi: 10.1609/aimag.v39i2.2782.

[36] Z. Ournani *et al.*, “Benchmarking To cite this version : HAL Id : hal-02403379 Taming Energy Consumption Variations in Systems Benchmarking,” 2020.

[37] D. Patterson *et al.*, “Carbon Emissions and Large Neural Network Training.” 2021, [Online]. Available: http://arxiv.org/abs/2104.10350.

[38] J. Ren, L. Gao, H. Wang, and Z. Wang, “Optimise web browsing on heterogeneous mobile platforms: A machine learning based approach,” *Proc. - IEEE INFOCOM*, 2017, doi: 10.1109/INFOCOM.2017.8057087.

[39] L. A. Tawalbeh, A. Basalamah, R. Mehmood, and H. Tawalbeh, “Greener and Smarter Phones for Future Cities: Characterizing the Impact of GPS Signal Strength on Power Consumption,” *IEEE Access*, vol. 4, pp. 858–868, 2016, doi: 10.1109/ACCESS.2016.2532745.

[40] T. Yang, Y. Chen, and V. Sze, “Designing Energy-Efficient Convolutional Neural Networks using Energy-Aware Pruning,” pp. 5687–5695.

[41] Z. Yu *et al.*, “Energy and performance trade-off optimization in heterogeneous computing via reinforcement learning,” *Electron.*, vol. 9, no. 11, pp. 1–14, 2020, doi: 10.3390/electronics9111812.

[42] M. Alioto, “From Less Batteries to Battery-Less Alert Systems with Wide Power Adaptation down to nWs -Towards a Smarter, Greener World,” *IEEE Des. Test*, vol. 2356, no. c, pp. 1–34, 2021, doi: 10.1109/MDAT.2021.3069087.

[43] P. E. Gill, W. Murray, and M. H. Wright, *Practical optimization*. SIAM, 2019.

[44] S. S. Rao, *Engineering optimization: theory and practice*. John Wiley \& Sons, 2019.

[45] A. Banerjee, L. K. Chong, C. Ballabriga, and A. Roychoudhury, “EnergyPatch: Repairing Resource Leaks to Improve Energy-Efficiency of Android Apps,” *IEEE Trans. Softw. Eng.*, vol. 44, no. 5, pp. 470–490, 2018, doi: 10.1109/TSE.2017.2689012.

[46] P. Probst and B. Bischl, “Tunability: Importance of Hyperparameters of Machine Learning Algorithms,” *J. Mach. Learn. Res.*, vol. 20, pp. 1–32, 2019, Accessed: Sep. 08, 2021. [Online]. Available: http://jmlr.org/papers/v20/18-444.html.

[47] J. Li *et al.*, “Feature selection: A data perspective,” *ACM Comput. Surv.*, vol. 50, no. 6, Dec. 2017, doi: 10.1145/3136625.

[48] “Feature Selection with Stochastic Optimization Algorithms.” https://machinelearningmastery.com/feature-selection-with-optimization/ (accessed Sep. 20, 2021).

[49] “Recursive Feature Elimination — Yellowbrick v1.3.post1 documentation.” https://www.scikit-yb.org/en/latest/api/model_selection/rfecv.html (accessed Sep. 20, 2021).

[50] B. Venkatesh and J. Anuradha, “A review of Feature Selection and its methods,” *Cybern. Inf. Technol.*, vol. 19, no. 1, pp. 3–26, 2019, doi: 10.2478/CAIT-2019-0001.

[51] M. Binder, J. Moosbauer, J. Thomas, and B. Bischl, “Multi-objective hyperparameter tuning and feature selection using filter ensembles,” *GECCO 2020 - Proc. 2020 Genet. Evol. Comput. Conf.*, pp. 471–479, 2020, doi: 10.1145/3377930.3389815.

[52] A. Noureddine, A. Bourdon, R. Rouvoy, and L. Seinturier, “Runtime monitoring of software energy hotspots,” *2012 27th IEEE/ACM Int. Conf. Autom. Softw. Eng. ASE 2012 - Proc.*, pp. 160–169, 2012, doi: 10.1145/2351676.2351699.

[53] B. Schäfer, C. Grabow, S. Auer, J. Kurths, D. Witthaut, and M. Timme, “Taming Instabilities in Power Grid Networks by Decentralized Control,” *Eur. Phys. J. Spec. Top.*, vol. 225, no. 3, pp. 569–582, Aug. 2015, doi: 10.1140/epjst/e2015-50136-y.

[54] V. Arzamasov, K. Bohm, and P. Jochem, “Towards Concise Models of Grid Stability,” *2018 IEEE Int. Conf. Commun. Control. Comput. Technol. Smart Grids, SmartGridComm 2018*, Dec. 2018, doi: 10.1109/SMARTGRIDCOMM.2018.8587498.

Appendix 1 – Sample pyRAPL Implementation Code

```
# Sample Implementation of pyRAPL (Training - Energy Measurement Assumed)
# ========================================================================
# Load ALL other relevant libraries

import pyRAPL # Import and setup pyRAPL
pyRAPL.setup()
meter = pyRAPL.Measurement('bar')

# Load the dataset, prepare and preprocess for the modeling to begin
# XGBoost classifier assumed and defined as xgb

meter.begin () # Start energy consumption measurement during training of the model from here
score = np.mean(cross_val_score(xgb, X_train, y_train, cv=splits, scoring='accuracy')) # Train and Evaluate the Model
meter.end() # End the energy consumption measurement of the block after training
train_cpu_energy = meter.result.pkg # cpu energy meter
train_dram_energy = meter.result.dram # dram energy meter

print( " CPU_Energy: " + str((train_cpu_energy)/1000000),
         " DRAM_Energy: " + str((train_dram_energy)/1000000)
# The division by 1000000 converts the energy from microJoules to Joules.
```

Appendix 2 – First 5 instances of the SGS Dataset

Out[2]:

	tau1	tau2	tau3	tau4	p1	p2	p3	p4	g1	g2	g3	g4	stab	stabf
0	2.959060	3.079885	8.381025	9.780754	3.763085	-0.782604	-1.257395	-1.723086	0.650456	0.859578	0.887445	0.958034	0.055347	unstable
1	9.304097	4.902524	3.047541	1.369357	5.067812	-1.940058	-1.872742	-1.255012	0.413441	0.862414	0.562139	0.781760	-0.005957	stable
2	8.971707	8.848428	3.046479	1.214518	3.405158	-1.207456	-1.277210	-0.920492	0.163041	0.766689	0.839444	0.109853	0.003471	unstable
3	0.716415	7.669600	4.486641	2.340563	3.963791	-1.027473	-1.938944	-0.997374	0.446209	0.976744	0.929381	0.362718	0.028871	unstable
4	3.134112	7.608772	4.943759	9.857573	3.525811	-1.125531	-1.845975	-0.554305	0.797110	0.455450	0.656947	0.820923	0.049860	unstable

Appendix 3 – First 5 instances of the Prepared SGS Dataset

Out[3]:

	tau1	tau2	tau3	tau4	p1	p2	p3	p4	g1	g2	g3	g4	stabf
0	2.959060	3.079885	8.381025	9.780754	3.763085	-0.782604	-1.257395	-1.723086	0.650456	0.859578	0.887445	0.958034	unstable
1	9.304097	4.902524	3.047541	1.369357	5.067812	-1.940058	-1.872742	-1.255012	0.413441	0.862414	0.562139	0.781760	stable
2	8.971707	8.848428	3.046479	1.214518	3.405158	-1.207456	-1.277210	-0.920492	0.163041	0.766689	0.839444	0.109853	unstable
3	0.716415	7.669600	4.486641	2.340563	3.963791	-1.027473	-1.938944	-0.997374	0.446209	0.976744	0.929381	0.362718	unstable
4	3.134112	7.608772	4.943759	9.857573	3.525811	-1.125531	-1.845975	-0.554305	0.797110	0.455450	0.656947	0.820923	unstable