

ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING APPROACH TO MEASURING ENERGY CONSUMPTION IN DATA CENTRE FACILITIES

SANJEEWA JAYATHILAKE



**A thesis submitted in partial fulfilment of the requirements of the
University of East London for the degree of Doctor of Philosophy**

October 2019

ABSTRACT

Data centres are at the heart of the modern digital world; However, at the same time it accounts for 10% of the world electricity supply. To improve energy efficiency, measuring energy consumption is an important step. However, it is a challenging task, especially in small to medium-sized data centres. Due to the setup of such facilities, it is not always feasible to measure the energy consumption (e.g. due to being positioned in mixed use buildings).

This research project addresses this problem by providing the tools and models to help estimate the energy consumption of data centres, with particular emphasis on smaller facilities. The work made two main novel contributions. First, energy models along with a web-based user-friendly tool were developed. The tool is capable of calculating energy consumption of each DC equipment type to then approximate the overall energy consumption of the facility. This tool is available for DC managers and operators. It uses publicly available benchmark data as input for the calculations.

One of the limitations of the first set of models is their reliance on pre-existing data for specific hardware. However, there are many ways hardware can be configured, meaning benchmark data was not available to all types of servers. As such, the second research contribution was the design of new machine learning algorithms capable of predicting energy consumption of servers based on a small number of features within 12% error rate. An open source software tool, **MALEP**, was also developed based on the machine learning algorithms to automate the prediction of the energy consumption of any servers, irrespective of the presence of benchmark data. The software is made available open source under the GNU General Public Licence and downloadable from GitHub.

Although this work focused on servers which account for the largest part of energy consumption in data centres, in future, we hope to extend this work to create such models for storage and networking equipment.

TABLE OF CONTENTS

ABSTRACT	i
TABLE OF CONTENTS	ii
LIST OF FIGURES	vii
LIST OF TABLES	xi
LIST OF EQUATIONS	xii
ACRONYMS	xiii
ACKNOWLEDGMENT	xiv
DEDICATION	xv
1 INTRODUCTION	2
1.1 Context & Motivation	4
1.2 Research Questions	5
1.3 Research Contributions	6
1.4 Thesis Outline	7
1.5 Research Workflow	10
2 LITERATURE REVIEW	11
2.1 Introduction	11
2.2 Problem Statement	11
2.3 Survey	13
2.4 Results Analysis	16
2.4.1 Filtered Results	16
2.4.2 Results breakdown	19
2.5 Detailed Analysis on Related Work	19
2.6 Critical Analysis Summary	24
2.7 Summary	25
3 MODEL BASED ESTIMATION	27
3.1 Introduction	27
3.2 Background on Energy Consumption	27
3.3 Energy Efficiency Metrics	28
3.3.1 The Green Grid Benchmarking Standards	29
3.4 Power Consumption Analysis	32
3.4.1 Categorization	32
3.5 Variable Definition	33

3.6 Models.....	34
3.6.1 Server.....	34
3.6.2 Storage.....	35
3.6.3 Network.....	36
3.6.4 UPS.....	37
3.6.5 Cooling.....	38
3.6.6 Total Power Consumption.....	38
3.7 Example.....	39
3.8 Energy Snapshots.....	42
3.9 Conclusion.....	44
4 MALEP v1: LINEAR REGRESSION PREDICTIONS	45
4.1 Introduction.....	45
4.2 Why Machine Learning in this research	45
4.2.1 Idle and fully utilized power consumption.....	46
4.3 Related Machine Learning Techniques.....	46
4.3.1 Unsupervised Learning.....	49
4.3.2 Supervised Learning.....	49
4.4 Linear Regression.....	50
4.4.1 Coefficient.....	51
4.4.2 Simple Linear Regression.....	51
4.4.3 Multiple Linear Regression	52
4.5 Server Attribute Analysis.....	53
4.5.1 Server Actual age Vs Reference age	53
4.5.2 Studied Attributes.....	54
4.5.3 Outliers Discovery and Removal.....	58
4.6 Single Linear Regression (SLR) Analysis	61
4.7 Multiple Linear Regression (MLR) Analysis	71
4.8 Conclusion on regression model analysis	75
5 MALEP v2: DEEP LEARNING PREDICTION.....	76
5.1 Introduction.....	76
5.2 What is Deep Learning?.....	76
5.2.1 Artificial Neural Networks.....	77
5.2.2 Deep Neural Networks	78
5.3 Why Deep Learning?	78
5.4 Related Deep Learning Techniques	78
5.5 Model Analysis	79
5.5.1 Feature selection.....	79
5.5.2 Models	80

5.6 Model Optimization	83
Model: Sequential.....	83
5.7 Summary	85
6 IMPLEMENTATIONS OF CULCULATION MODELS	87
6.1 Introduction	87
6.2 Application Architecture.....	87
6.2.1 Other Design Patterns.....	88
6.3 Technology.....	89
6.3.1 Java & EcoSystem.....	89
6.3.2 VAADIN	90
6.3.3 Maven.....	90
6.3.4 Version Controlling & Code Repository.....	92
6.3.5 Environment Management	92
6.3.6 Unit Testing and Test-Driven Development	93
6.3.7 Deployment	95
6.4 Workflow of the Tool.....	95
6.4.1 Energy Calculator.....	95
6.4.2 Output Screen	96
6.4.3 Energy Snapshots	97
6.5 Application Security.....	103
6.6 Code Samples.....	103
6.7 Model Evaluation	103
6.7.1 Server Power Consumption Calculation	104
6.7.2 Storage Power Consumption Calculation.....	104
6.7.3 Network Power Consumption Calculation.....	105
6.7.4 UPS Power Consumption Calculation.....	105
6.7.5 Cooling Power Consumption Calculation	105
6.8 Conclusion.....	106
7 IMPLEMENTATIONS OF MALEP	107
7.1 Introduction	107
7.2 Design & Architecture	107
7.2.1 Overall View	107
7.2.2 Microservices for Integration	109
7.3 Technology.....	110
7.3.1 Python & EcoSystem.....	110
7.3.2 Neural Network	111
7.3.3 Virtual Environment.....	114
7.4 Restful Webservices.....	115

Security.....	115
Sample JSON Request.....	115
Web Services Code.....	116
7.5 Micro Framework and API development.....	117
7.6 Challenges faced	118
7.7 Summary	119
8 EVALUATION OF MALEP: REGRESSION MODELS.....	121
8.1 Test Data	121
8.1.1 Obtained data set	121
8.1.2 Data Cleansing and Feature Selection.....	122
8.1.3 Test Data Analysis.....	123
8.1.4 Statistical Details of Full Data Set.....	127
8.1.5 Test Data Utilization Strategy	127
8.1.6 Statistical Analysis of Training Data Set.....	128
8.1.7 Statistical Analysis of Test Data Set	128
8.2 Regression Model Evaluation	129
8.2.1 Model Evaluation Criteria	129
8.2.2 Evaluation of Fully Utilized Energy Consumption.....	130
8.2.3 Evaluation of Idle Power Consumption	144
8.2.4 Result Comparison of Regression Evaluation.....	153
8.2.5 Conclusion on Regression Result Analysis.....	155
8.3 Deep Learning Model Evaluation	156
8.3.1 Performance Tuning	156
8.3.2 Evaluation of Fully Utilized Power Prediction	161
8.3.3 Evaluation of Idle Energy Consumption	169
8.3.4 Results Comparison - Power Consumption.....	173
8.3.5 Analysis of Results.....	175
8.4 Summary	175
Conclusion.....	177
9 CONCLUSION AND FUTURE WORK	178
9.1 Summary and Conclusion	178
9.2 Future Perspectives	180
9.2.1 Short Term.....	180
9.2.2 Long Term.....	180
REFERENCES.....	182
APPENDICES.....	194
Appendix 1: Code Samples.....	194

1 Calculation Model Web Tool	194
2 MALEP	197

LIST OF FIGURES

<i>Figure 1: Structure of the Thesis</i>	8
<i>Figure 2 :Research methodology workflow</i>	10
<i>Figure 3 - Literature survey steps</i>	14
<i>Figure 4:Results Breakdown</i>	19
<i>Figure 5:Energy usage in DC</i>	29
<i>Figure 6:Breakdown of Consumption</i>	31
<i>Figure 7:Snapshot organization</i>	43
<i>Figure 8:Workflow of Snapshots</i>	44
<i>Figure 9:Machine Learning Workflow</i>	47
<i>Figure 10:ML categorization</i>	48
<i>Figure 11:processor speed vs predicted energy consumption for selected dataset</i>	62
<i>Figure 12:CPU speed vs energy consumption for the whole data set</i>	63
<i>Figure 13:Actual Vs Predicted</i>	64
<i>Figure 14:Actual Vs Predicted</i>	65
<i>Figure 15:Actual Vs Predicted</i>	66
<i>Figure 16:Actual Vs Predicted</i>	67
<i>Figure 17:Actual Vs Predicted</i>	68
<i>Figure 18:Actual Vs Predicted</i>	69
<i>Figure 19:Actual Vs Predicted</i>	70
<i>Figure 20:Actual Vs Predicted</i>	70
<i>Figure 21:Actual Vs Predicted</i>	72
<i>Figure 22:Actual Vs Predicted</i>	73
<i>Figure 23:Actual Vs Predicted</i>	74
<i>Figure 24 : Deep Learning</i>	77
<i>Figure 25 : Neural Network Structure</i>	78

<i>Figure 26: Organization of a Neural Network</i>	79
<i>Figure 27 : MALEP Neural Network Layers</i>	81
<i>Figure 28 : Visualized Neural Network Model</i>	82
<i>Figure 29 : Dataflow of the Model</i>	83
<i>Figure 30 : Sequential Model Actual vs Predicted</i>	85
<i>Figure 31: Application Architecture</i>	87
<i>Figure 32 :Multi-threaded sessions</i>	89
<i>Figure 33:Code Promotion Steps</i>	93
<i>Figure 34 :TDD Cycle</i>	94
<i>Figure 35 :Input Screen</i>	96
<i>Figure 36 :Output Screen</i>	97
<i>Figure 37 Energy Snapshot</i>	98
<i>Figure 38 :Snapshot Table Structure</i>	99
<i>Figure 39 :Server Energy</i>	99
<i>Figure 40 : Server Energy Table Structure</i>	100
<i>Figure 41 :Network Energy</i>	100
<i>Figure 42 :Network energy Table Structure</i>	101
<i>Figure 43 : Storage Energy</i>	101
<i>Figure 44: Network Energy Table Structure</i>	102
<i>Figure 45 :Cooling Energy</i>	102
<i>Figure 46 : Cooling energy table structure</i>	102
<i>Figure 47:Energy Calculation Results</i>	103
<i>Figure 48:Evaluation of Server Power consumption</i>	104
<i>Figure 49:Evaluation of Storage s Power consumption</i>	104
<i>Figure 50:Evaluation of Network Power consumption</i>	105
<i>Figure 51:Evaluation of UPS Power Consumption</i>	105
<i>Figure 52:Evaluation of Cooling Power Consumption</i>	105

<i>Figure 53:Architecture of the Integrated System</i>	108
<i>Figure 54:Application Integration</i>	110
<i>Figure 55:MALEP NN</i>	113
<i>Figure 56 : MALEP Virtual Environment</i>	114
<i>Figure 57 :JSON Request</i>	115
<i>Figure 58 : JSON Response</i>	116
<i>Figure 59:MALEP Microframework</i>	118
<i>Figure 60:A sample server specification with energy consumption data from SPEC</i>	121
<i>Figure 61 :Cleansed training data sample in tabular format</i>	123
<i>Figure 62 : Distribution of fully utilized power</i>	124
<i>Figure 63:Distribution of idle power</i>	124
<i>Figure 64:Distribution of Processor Speed</i>	125
<i>Figure 65:Distribution of memory</i>	125
<i>Figure 66:Distribution of number of cores</i>	126
<i>Figure 67:Distribution of reference age</i>	126
<i>Figure 68:Test Data Strategy</i>	127
<i>Figure 69:Predicted Vs Actual</i>	132
<i>Figure 70:Actual Vs Predicted</i>	133
<i>Figure 71:Predicted Vs Actual -Scenario 2</i>	135
<i>Figure 72: Actual Vs Predicted</i>	135
<i>Figure 73:Predicted Vs Actual - Scenario 3</i>	137
<i>Figure 74:Actual Vs Predicted</i>	138
<i>Figure 75 :Actual Vs Predicted</i>	139
<i>Figure 76:Actual Vs Predicted</i>	140
<i>Figure 77:Actual Vs Predicted</i>	141
<i>Figure 78 :Actual Vs Predicted</i>	142
<i>Figure 79:Actual Vs Predicted</i>	143

<i>Figure 80:Actual Vs Predicted</i>	144
<i>Figure 81:Actual Vs Predicted</i>	145
<i>Figure 82:Actual Vs Predicted</i>	147
<i>Figure 83:Actual Vs Predicted</i>	148
<i>Figure 84:Actual Vs Predicted</i>	150
<i>Figure 85:Actual Vs Predicted</i>	150
<i>Figure 86:Actual vs Predicted</i>	152
<i>Figure 87:Actual Vs Predicted</i>	153
<i>Figure 88:Model Accuracy</i>	158
<i>Figure 89:Model Accuracy</i>	159
<i>Figure 90:Model Accuracy</i>	160
<i>Figure 91 : Actual vs Predicted – Scenario 1</i>	162
<i>Figure 92:Actual vs Error – Scenario 1</i>	162
<i>Figure 93:Model Accuracy</i>	163
<i>Figure 94:Actual vs Predicted – Scenario 2</i>	164
<i>Figure 95:Actual vs Error – Scenario 2</i>	164
<i>Figure 96 : Model Accuracy</i>	165
<i>Figure 97:Actual Vs Error Percentage - Scenario 3</i>	166
<i>Figure 98:Actual Vs Predicted - Scenario 3</i>	166
<i>Figure 99:Model Accuracy Vs Epoch – Scenario 3</i>	167
<i>Figure 100:Actual Vs Predicted - Scenario 4</i>	168
<i>Figure 101 : Actual Vs Error Percentage - Scenario 4</i>	168
<i>Figure 102:Model Accuracy - Scenario 4</i>	169
<i>Figure 103:Actual Vs Predicted</i>	171
<i>Figure 104:Actual Vs Error percentage – Scenario 1</i>	171
<i>Figure 105:Architecture of proposed Recommendation System</i>	181

LIST OF TABLES

<i>Table 1 : Research Questions</i>	6
<i>Table 2 : Information Sources</i>	14
<i>Table 3 : Exclusion Criteria</i>	15
<i>Table 4 results breakdown: Search Results</i>	18
<i>Table 5 : Results breakdown</i>	19
<i>Table 6:Energy Consumption Example</i>	30
<i>Table 7 : Test Data Attributes</i>	57
<i>Table 8 :Java Frameworks used in the Application</i>	90
<i>Table 9 :Application Environments</i>	92
<i>Table 10:Python Frameworks Used</i>	111
<i>Table 11 : Cleansed training data sample</i>	122
<i>Table 12 : Summary of Full data set</i>	127
<i>Table 13 : Training Data Analysis</i>	128
<i>Table 14:Test Data Analysis</i>	129
<i>Table 15:Test Data Summary</i>	129
<i>Table 16:Prediction Results Comparison</i>	154
<i>Table 17:Scenario 1 Results</i>	167
<i>Table 18 : Scenario 1 Results</i>	170
<i>Table 19:First 10 Predictions</i>	170
<i>Table 20:Results Comparison: Fully Utilized Power</i>	174

LIST OF EQUATIONS

<i>Equation 1: Total DC Energy Consumption</i>	27
<i>Equation 2: PUE</i>	29
<i>Equation 3: DCiE</i>	31
<i>Equation 4: Total Server Energy</i>	35
<i>Equation 5 : Total Storage Energy</i>	36
<i>Equation 6 : Total Network Energy</i>	36
<i>Equation 7: Total UPS Energy</i>	37
<i>Equation 8 : Total Cooling Energy</i>	38
<i>Equation 9: Total Energy of all equipment</i>	38
<i>Equation 10: Simple Linear Regression</i>	51
<i>Equation 11 : Multiple Linear Regression</i>	52
<i>Equation 12 : Multiple Linear Regression</i>	71
<i>Equation 13 : Error Percentage Calculation</i>	130

ACRONYMS

DC	Data centre
ML	Machine Learning
AI	Artificial Intelligence
SLR	Simple Linear Regression
MLR	Multi Linear Regression
DL	Deep Learning
NN	Neural Network
ANN	Artificial Neural Network
DNN	Deep Neural Network
UPS	Uninterrupted Power Supply

ACKNOWLEDGMENT

Firstly, and foremostly, I must thank my supervisor, Dr Rabih Bashrouh for his support, guidance and encouragement. Without him, I would not certainly have come this far. Throughout my PhD journey, he has been very supportive and selflessly spent his time every time I was looking for his help.

I would also like to thank my secondary supervisor Dr Saeed Sharrif.

And then number of the academic staff in the department of Computer and Architecture of the University of East London should also be mentioned for their valuable time in reviewing my work and providing feedback.

A special thanks to Richard Bottoms, Charlotte Forbes and Avinder Bhinder from non-academic department for their help in various administrative tasks.

Lastly, but not least, I should thank my family for their love and patience. My wife Wayani and three sons – Sanuka, Sithuka and Sumika have always supported and encouraged me during last few years. Most evenings and weekends were spent on doing research, when I should have been with them.

DEDICATION

To my wife. Without her continuous support and encouragement, I would not have come this far.

Part I

Introduction & Literature

“Twenty years from now you will be more disappointed by the things that you didn’t do than by the ones you did do.”

- **Mark Twain**

INTRODUCTION

Measuring power consumption is a challenging task due to the lack of affordable measuring units and the inherently complex nature of infrastructure setup in the data centres. The way it is setup, it is not practical to switch off the centre and install measuring units at equipment level.

Some of these DCs are operated in the mix of an office environment sharing the power supply with other operations like workstations, printers and even kitchen appliances. This makes it harder to isolate the actual power consumed by the DC equipment. In relation to such a mixed-up environment, there is a motivational aspect as well. The bills may be paid for the whole setup, hence there may not be enough pressure to be more energy efficient. So, there is no drive for measuring the actual power utilized by the DC.

Whatever the reason, if it is not feasible to measure the power consumption physically, we need to consider alternative approaches. And we chose to estimate it through modelling and prediction.

This research project identifies the existing attempts and limitations in modelling and predicting the equipment level energy consumption in data centre facilities. One major limitation in existing attempts is that they are focusing only on CPU energy models, not the overall features of servers. Lack of availability of these models in the form of enterprise tools for data centre operators to model and quantify their equipment level consumption is another limitation.

We investigated how best to address the limitations in the existing models and techniques. As a result, we developed models which are capable of assisting the data centre operators, managers and other stakeholders. The models help to calculate both total and break down energy consumption of IT and infrastructure. They also help in calculating other benchmarking matrices like PUE. This information enables DC stakeholders to make informed and effective decisions in energy efficiency of their respective portfolios. Analysis and details of the improved models are discussed in chapter 3.

We then built a complete web tool to implement models discussed in chapter 3 and integrated it as a new module into the existing Eureka application. Data centre operators can login to the Eureka application and create their energy portfolio. They can then use the new component to calculate the energy consumption of each equipment category and the total number. The new tool implementation and evaluation details are presented in chapter 6.

The models discussed above rely on some input parameters like fully utilized and idle power consumption of each individual equipment. While it is not difficult to find out this information for standard specifications, more often than not, data centres use custom built equipment. It is not realistically possible to find out or calculate those consumption values for such specifications. This challenge presented us the research question of how to predict the power consumption of servers using publicly available and reliable data. While it would be beneficial to apply this prediction into all sorts of equipment, we only focused on servers to perform our analysis, implementation and result benchmarking. The main reason for choosing only servers was because they are by far the biggest energy consuming IT equipment in a data centre facility. Also, compared to other equipment like storage or network, it is very common for servers to be built with custom specifications.

We looked into a number of approaches of solving this problem, but machine learning looked to be the best technique. The main reason was the availability of reliable data for known specifications in the public domain. Machine learning is very much data driven and renowned for its ability in predictions with the existing data to feed training models.

A general overview of related Machine Learning techniques followed by comprehensive model analysis using Multi Linear Regression and Deep Learning are presented in chapter 4 and 5.

The best way to evaluate models is by implementing a proof of concept application. So, we built an open-source enterprise tool, incorporating the ML regression and deep learning models. We called it MALEP (Machine Learning Energy Predictor) and made it available to download from a well-known public code sharing repository GitHub under <https://github.com/jaysanjeewa/MaLEP>

Implementation details of the MALEP tool is discussed in chapter 7. The models were thoroughly evaluated by using multiple scenarios with real-world test data. We

achieved the predicted energy consumption with 12% of error percentage in the best-case scenarios.

The conclusion and the future work are discussed at the end.

1.1 Context & Motivation

With more and more devices connected to the internet, services offered over the internet are increasingly becoming affordable. This is mainly due to the fact that these services are provided by the shared resources which are hosted in data centres. When the resources are shared, the cost of the services tends to be low.

Data centres provide scalable infrastructure solutions at an affordable price for both organizations and individuals. While they enable complex and distributed computer systems to become highly available (HA), they also simplify the platform solution capability.

Being one of the highest energy consumers to power the infrastructure and processing capability, a major challenge data centres are facing today is improving their energy efficiency. 57% of computers in the US are housed in small data centres [*Cheung, 2014*]. Most of the big data centres are usually operated by corporate companies. For them energy efficiency is the main objective in achieving operational cost saving targets. Small to medium-size operators are isolated and not so motivated in achieving a high level of efficiency.

Among all other factors, the most important aspect in making these DCs more energy efficient is measuring the energy consumption. Without being able to measure actual consumption, it is hard to find out where to look for improvements.

So, the question becomes if all data centres are capable of measuring their energy consumption?

The simple answer is “no”.

Due to the high complexity of the setup, measuring the energy consumption of the DC is not a straightforward process, especially for small to mid-sized operators.

While the high cost of measuring hardware technology is the main reason, regulations and data security are some other reasons which contribute to make energy consumption measuring a challenging piece of work. These reasons are again more of a challenge for smaller to medium-sized data centres.

In this project we researched how to identify models to profile the energy consumption for each category like servers, storage and networking devices. We then built a proof of concept tool to implement and test those models and then to validate the results.

Servers are the biggest energy consumers in DCs and typically accounts for about 56% of total power consumption [Pelley et al, 2009]. The total energy consumption models we developed and implemented for servers need some input parameters which are not straight forward to find out. Idle and fully utilized server power consumption are such attributes.

Usually these standard energy consumption rates for well-known specifications are published and available online. However, most of the time servers in the DCs are not into these well-known specs, hence it is very difficult to know their energy consumption rates.

So, we directed the research into predicting the energy consumption of the servers by leveraging machine learning techniques.

We developed a second tool MALEP, to train ML model on published server specifications and then to predict the energy consumption for new or unknown specifications.

1.2 Research Questions

The biggest challenge of this project is the lack of knowledge on the actual power consumption on small to mid-sized data centres. So, the first step is identifying the existing attempts on this front. Analysis on such research work helps the understanding of the research context as well as identifying the drawbacks of them to decide where to direct our research.

If for any reason, it is not feasible to physically measure the power consumption, how can we estimate it? The research work on developing required models and algorithms in helping this approximation plays an integral role in this.

Having identified the models, how can they be made available for the real-world data centre stakeholders to calculate their respective consumption.

In line with that, table 1 lists the research questions which are investigated in this project.

RQ1	what are the current measures & methods used to measure energy consumption?
RQ2	How can we estimate the energy consumption in data centres without physically measuring it?
RQ3	How can we automate the energy calculation process?

Table 1 : Research Questions

This thesis proposes an approach to formally answering these research questions.

1.3 Research Contributions

The main contribution of this research is a novel approach to measuring the energy consumption in the data centre facilities.

First by modelling the variables of each equipment types and equations for estimating the total energy consumption. Then for idle and utilized power consumption of servers, AI algorithms were extensively leveraged to predict each aspect of consumption. Each new research contribution is as below.

1. Models for approximately calculating equipment category level and the total energy consumption in data centres.

2. A web tool implementing above models and equations for data centre operators and managers to calculate the energy consumption in their DCs. The tool also provides them the capability of estimating the potential energy efficiency of any changes to the equipment profile.
3. A set of AI algorithms based on linear regression, deep learning and neural networks in predicting the energy consumption of servers with uncommon specifications.
4. Open Source Product: MALEP, A tool developed by using modern architectural patterns and technology to incorporate models and algorithms.

1.4 Thesis Outline

This thesis is organized into nine chapters. A specific aspect of the research work is discussed and presented in each chapter.

Figure 1 illustrates the general overview of the structure:

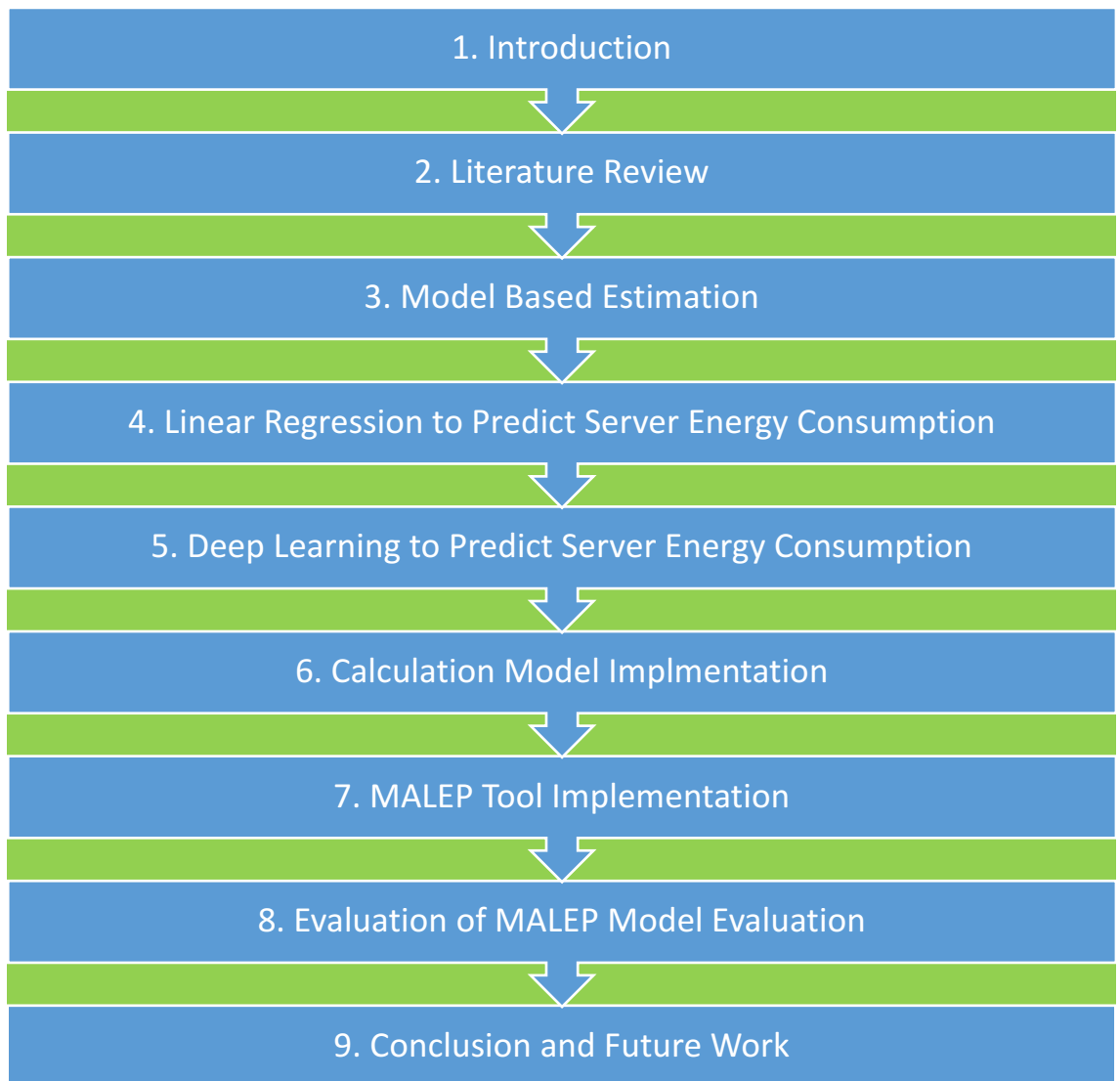


Figure 1: Structure of the Thesis

Chapter 1 presents the insights into the research background, motivation and research questions. It also highlights individual research contributions.

Chapter 2 presents the literature review on existing attempts of measuring the energy consumption. We searched several information sources for primary research and extracted key information, along with metadata. A background on terminology and concepts with an example is provided in the latter part of this chapter.

Chapter 3 starts with an analysis on power consumption, outlining the equipment types, along with the attributes studied under this research. It describes the energy model calculator for each type of data centre equipment. Under each section, the related equation and variable definitions are also presented.

Chapter 4 starts with the research questions related to machine learning and justification on why AI and ML were chosen to solve some of the research problems. We then discuss the related ML techniques. Moving onto the model analysis, details on different types of regression models applied with examples presented. At the end, we discuss how we leveraged ML to successfully predict the energy consumption of servers and how they were executed on the test data.

Chapter 5 As seen in chapter 4, the results in predicting server energy consumption using linear regression models can be further improved. So, we turned the focus into deep learning and discussing the basics of the DL and why DL was chosen for predicting server energy consumption. Model analysis including feature selection decisions are also presented at the latter part of the chapter.

Chapter 6 focuses on the implementation and evaluation of the energy calculator web tool. Discussion on the design patterns and the technology stack chosen is followed by the application workflow design.

Chapter 7 discusses the details of the MALEP tool implementation by outlining the architecture and the technology. How Python and related libraries were used to develop neural network models as a pluggable prediction engine is discussed with the illustrations. We then explain how we designed and developed the out of the box integration for external application. Microservices and the related restful web services we built for the integration are explained with usage examples. The other usage channels like desktop client and the web-based tool offering is also discussed with examples.

Chapter 8 evaluates MALEP using a wide variety of scenarios. Analysis on the evaluation results on regression and deep learning models, followed by a results comparison for both regression and deep learning models are detailed. We present the conclusions drawn based on the result analysis at the end.

Chapter 9 concludes the research work. Both short term and long-term future work are discussed.

1.5 Research Workflow

We first reviewed the existing work and then extracted the results. The extracted features were analysed and synthesized for better understanding of the research problems. Details and critical summary on the existing literature review are presented. A detailed analysis was carried out to fine-tune the research questions and to guide the research towards addressing the refined research questions.

The next stage was to study the existing models and understand how to align our research to come up with new models in estimating the energy consumption of each equipment type in the data centres.

After carrying out a gap analysis on the existing approaches, we proposed improved new models. They were implemented and evaluated using real world benchmark data. At the model evaluation stage, it was evident that some of the model attribute values like idle and utilized power consumption not available for most of the server models in the data centres. To address this problem, we directed research into artificial intelligence and machine learning.

Both linear regression and deep learning models were studied, analysed and implemented as enterprise applications. Those models were evaluated using publicly available benchmark data to test the efficiency of the MLL models.

Figure 2 shows the research methodology as in a workflow diagramme.



Figure 2 :Research methodology workflow

Chapter II

LITERATURE REVIEW

2.1 Introduction

This chapter has the details on the review of existing literature in both academia and the industry in the context of estimating energy consumption in data centres. Firstly, basic terminology is explained, followed by the context and the existing work related to this research. Details on energy efficiency matrices and how they are utilized to benchmark the results are also explained.

The main purpose of this section is to present our effort in identifying and rationalizing the existing research in the area of energy consumption measuring in data centres, in particular at the equipment level. Due to the inherent complexity in the equipment organization and setup, realistically it is very difficult to measure the consumption physically. So, our attention was directed towards the publications in the area of energy consumption modelling and estimation.

2.2 Problem Statement

With more and more devices connected to the internet, services offered over the internet are increasingly becoming affordable. It was in 2008 that the number of devices connected to the Internet exceeded number of people living in the world [Evans, 2011]. These devices may not necessarily be computers, smartphones or tablets. They even include monitoring devices which are attached to the farm animals. [Avgerinou, 2017] discusses the details of a survey of data centres with their PUE values.

There is an increasingly big demand for the cloud hosted services and infrastructure on the cloud. These cloud-based services provide infrastructure, platform and application software which are much cheaper than running them on your own. The facilities where these services are hosted are called data centres. Depending on the size, they run a number of pieces of IT equipment like computer servers, storages, network equipment and need the infrastructure services like cooling mechanisms which are used to control the heat

within the facility. There are some other utility services like lighting which also fall into the category of infrastructure.

Data centres and server room facilities are the backbone of digital economy. They provide the digital infrastructure and the connectivity for public, corporate and small business users. Opting for such services enables business users to focus on solving their business problems rather than spending time and effort on basic technical work like platform set up and maintenance.

Pooling and sharing the IT resources has enabled the data centres to provide their service at an increasingly affordable price. They also employ software-based sharing techniques like virtualization and offer cloud-based solutions for infrastructure, services and application.

Server rooms in a data centre usually consist of computer servers, storage devices, network equipment and cooling mechanisms such as air conditioners. Depending on the utilization strategy, the power consumption of each category varies.

All of this equipment is powered by electricity. Even in small and mid-sized enterprises (SMEs), the power and cooling of data centre equipment can easily cost tens or hundreds of thousands of dollars a year. Data centres are energy hungry and it is estimated that 20% of the budget of a typical DC is on energy consumption [Infosys].

The carbon footprint created by data centres is a major concern and the total of the entire world is believed to be more than the total aviation industry [Bashroush, 2016]. In 2013, US data centres consumed an estimated 91 billion kilowatt-hours of electricity which is equivalent to the production of 34 large coal-fired power stations. This was expected to increase to 140 billion annually in 2020 [14]. About 2% of the global CO₂ emission is by data centres and the overall IT sector, [Avgerinou, 2017] which is a big concern. With the ever-increasing demand for IT processing power worldwide, this trend will only move upwards.

Data centres consume extreme amounts of high energy, so their environmental footprint is considerably high. So, it is very important to identify the areas to improve efficiency and reduce power consumption

However not every data centre is large scale and well-equipped with monitoring provisions. Small to medium data centres have not got the capability to measure their individual energy consumption and environmental footprint.

In some industries, economic and environmental pressure forces the small players to integrate with large data centres which are already equipped with measuring the consumption and applying the efficiencies. However, due to various reasons, this trend is not happening in other areas where operating small data centres is still the best practical solution. That is why it is very important to research on improving efficiency on smaller to mid-sized data centres.

A number of initiatives are already in place to improve the efficiency of the server room facilities. More energy efficient cooling units and energy saving building design are some of the examples

In the process of improving energy efficiency, measuring the energy consumption of a data centre is the fundamental requirement. Large data centres use modern measuring equipment but most of the time, this is not feasible for small to mid-sized data centres due to various reasons. While the unaffordable cost of those measuring units is the main reason, things like un-supported infrastructure and unadaptable legacy infrastructure are some of the other reasons.

One solution to overcome this barrier is to look into alternative ways. Estimating via the modelling energy consumption is one way of doing it. This research mainly focuses on modelling energy consumption and testing, validating them to measure small to mid-sized data centres.

2.3 Survey

To examine the existing work on this area, we conducted a survey on published literature measuring the data centre energy consumption. After finalizing the keywords, a search on the well-known research information sources was performed. The retrieved results were put through a filtering process to select only the relevant ones and then analysis was carried out on. At the end, conclusions were made based on the outcome of analysis.

Figure 3 below presents the steps of the literature survey:



Figure 3 - Literature survey steps

2.3.1 Keywords and search queries

The quality and the relevance of the results of a survey depend on the keywords. The list of keywords used in the search queries are listed below.

- Data centre
- Energy Consumption OR Energy Usage
- Estimation OR Modelling OR Predicting

2.3.2 Information Sources

We searched in the well-known information sources for published literature. Table 2 lists down the information sources used.

Source	Reference
IEEE Explore	https://ieeexplore.ieee.org
Google Scholar	https://scholar.google.com
ACM Digital Library	https://dl.acm.org
Semantic Scholar	http://semanticscholar.org

Table 2 : Information Sources

2.3.3 Exclusion Criteria

To study more relevant and related work, a filtering of the results was carried out by applying an exclusion criterion on the results obtained by the search. Table 3 lists down the exclusion criteria.

Criteria Number	Criteria
1	<u>Publication year outside of 1990 and 2017</u>
2	<u>Publication not in English</u>
3	<u>Publication is too short or too long</u> threshold was set as - No less than 2500 words - No more than 7000 words
4	<u>No Concise abstract, summary, conclusion.</u> Otherwise it is hard to understand the models, or the techniques applied in measuring or estimating the consumption.

Table 3 : Exclusion Criteria

Not many researches before 1990 have been published, mainly due to the internet was not available during that time. 2017 was the year this review was conducted, hence why the upper limit. Due to the practical reasons, the language chosen of this research was English, so any publication not in English was excluded. When the publication is too short, it is not easy to draw any conclusion out of it. On the hand, when it is too long, it is equally difficult to read all of it. Abstract, summary or the conclusion are the areas where the publication is summarised. When such areas are not concise or clear, it is difficult to understand the context, or the research methodologies applied.

2.4 Results Analysis

2.4.1 Filtered Results

Table 4 lists down the filtered results after applying the exclusion criteria

Reference	Title	Author	Year	Source
R1	Accurately Predicting the Energy Consumption of Your Data Centre	P. Bemis	2012	Google Scholar
R2	Measuring and Analyzing Energy Consumption of the Data Centre	T. Makris	2017	Semantic Scholar
R3	Power prediction for Intel XScale/spl reg/ processors using performance monitoring unit events	G. Contreras, M. Martonosi	2005	IEEE Explorer
R4	Complete System Power Estimation Using Processor Performance Events	W.L Bircher, L.K John	2012	IEEE Explorer
R5	Full-System Power Analysis and Modelling for Server Environments	D. Economou et al.	2006	Semantic Scholar
R6	Analysis of an Energy Proportional Data Centre	R. Lent	2015	ACM DL

R7	Run-time Energy Consumption Estimation Based on Workload in Server Systems	Lewis et.al	2008	ACM DL
R8	Real Time Power Estimation and Thread Scheduling via Performance Counters	K.Sing, M. Bhadauria	2009	ACM DL
R9	Run-time modelling and estimation of operating system power consumption	T. Li, L.K. John	2003	ACM DL
R10	RAPL: Memory power estimation and capping	H. David et al.	2010	IEEE Explorer
R11	Energy Conservation in Heterogeneous Server Clusters	T. Heath et al	2005	ACM DL
R12	Measuring Server Energy Proportionality	C. Hsu	2015	ACM DL
R13	Power monitors: a framework for system-level power estimation using heterogeneous power models	T. Bansal et al.	2005	IEEE Explorer

R14	A Methodology to Predict the Power Consumption of Servers in Data Centres	R. Basmadjian et al.	2011	ACM DL
R15	Distributed dynamic speed scaling	R Stanojevic, R Shorten	2010	Google Scholar
R16	Cutting the Electric Bill for Internet-Scale Systems	Qureshi et al.	2009	ACM DL
R17	D-Pro: Dynamic Data Center Operations with Demand-Responsive Electricity Prices in Smart Grid	P Wang et al.	2012	IEEE Explorer
R18	A Comparison of High-Level Full-System Power Models	S Rivoire et al	2008	ACM DL
R19	Minimizing Electricity Cost: Optimization of Distributed Internet Data Centers in a Multi-Electricity-Market Environment	L Rao	2010	IEEE Explorer
R20	Server selection for carbon emission control	Doyle et al.	2011	Google Scholar

Table 4 results breakdown: Search Results

2.4.2 Results breakdown

Table 5 lists the number of results retrieved.

Information Source	Number of results
IEEE Explore	6
Semantic Scholar	2
ACM Digital Library	9
Google Scholar	3

Table 5 : Results breakdown

Pie chart below visualises the results:

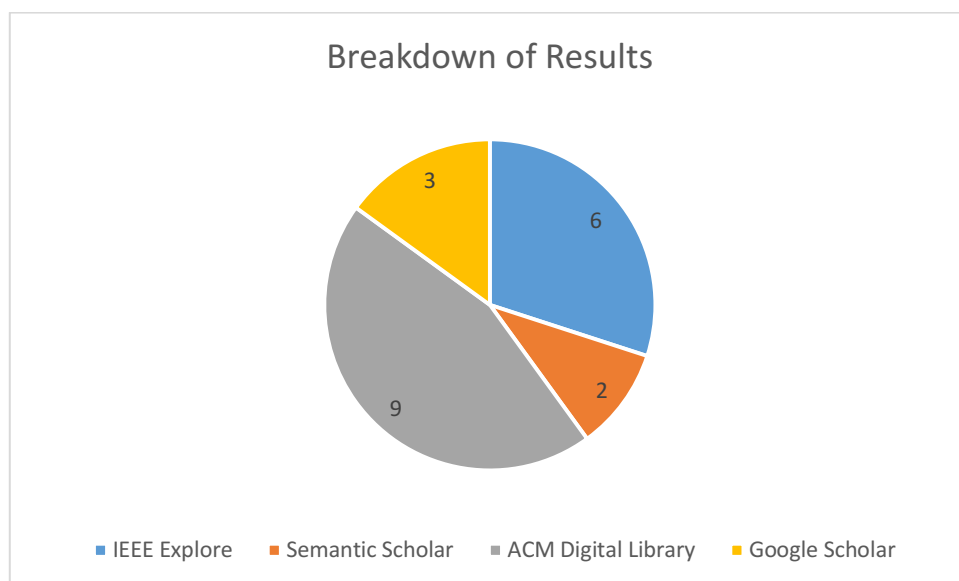


Figure 4:Results Breakdown

2.5 Detailed Analysis on Related Work

We studied the literature retrieved in the survey (section 2.3) and other related work. As a result of that effort we present the analysis on them below.

Measuring actual energy consumption is the most important aspect of analysis on energy footprint. Measuring involves both energy available and consumed. One way of doing this is by modelling resource usage in both hardware and software level. [Noureddine, Rouvoy and Seinturier, 2013] is a review of such energy measurement approaches.

System-level metrics are used in modelling processor power consumption by considering the correlation between power consumption and application execution. [Bircher and John, 2007]

Hardware support for virtualization is another aspect as it is a major mechanism used in data centres to deliver low-cost services. Virtualization helps to deliver abstract independent server units by utilizing the same hardware underneath. How to measure and apply the energy efficiencies in this area is discussed in detail in [Newcombe, Data centre energy efficiency metrics]

When it comes to measuring the energy consumption of software applications, it is broken into Active, Waiting and Idle consumption. A software application is not used 24 X 7 even though it may have a non-functional requirement to make it available all the time. Analysis of energy efficiency in a software application is discussed in [Noureddine, Islam, and Bashroush, 2016]

Using hardware performance counters to estimate run time energy consumption is discussed in [Lewis, Ghosh, and Tzeng, 2008]. Data centres usually over-provision the power supply to deal with unexpected scenarios. This results in under-utilization in capacity. Due to this, it is important to quantitatively identify the relationship between power consumption and system-level thermal load to optimize the power capacity.

IT equipment utilization percentage is identified as an important aspect of energy efficiency. When a piece of equipment is highly utilized, it runs at a higher energy efficiency rate.

Green Grid is a non-profit consortium (www.thegreengrid.org) that works to improve IT and data centre resource efficiency around the world. They have developed a number of

metrics to help with achieving this. ICT Capacity and Utilization Metrics [Green Grid ICT-Capacity-and-Utilization-Metrics, 2017] is one of them and it deals with ICT device utilization.

Data centre maturity model [The Green Grid Data Centre Maturity Model, 2011] by Green Grid introduces capability descriptors to help data centre users to benchmark current performance. This helps to identify the improvements and innovations to achieve higher efficiencies.

A lot of work has been done in the area of how to reduce the energy consumption while offering the same computation power in the data centre.

Electricity cost per computational unit varies from one geographical location to another. The option of exploiting this for advantageous purposes using distributed systems is discussed in [Qureshi et al., 2009]. One option explained is the usage of modern distributed systems to get the most computations done in the data centres where electricity is cheaper. How to optimize a distance-constrained energy price is another aspect discussed in it and conclude that even existing system may be able to save millions of dollars a year in electricity costs. The model the systems energy consumption of clusters as being proportional and near linear.

An algorithmic approach to run processors at dynamic speed scaling is discussed in [Stanojevic and Shorten, 2010]. They propose low-overhead fully decentralized algorithms which adjust the processor speed based on the load to solve the problem of non-linear relationship between energy consumption and the performance. Simulations based evaluation is used to measure the efficiency of the optimal solution. The adoption of these algorithms indicates a possible cost reduction between 10 – 40%.

An algorithm which formulates the electricity of the cloud as a flow network to minimize energy cost is presented in [Rao et al., 2010].

An algorithm which uses corrected marginal cost algorithms to minimize electricity cost is presented in [Wang, 2012].

An algorithm which minimizes the costs function of the carbon intensity and average job time is presented in [Doyle, O'Mahony, and Shorten, 2011]. They used a traffic generator to simulate the performance of the algorithms. Their results imply that carbon emission can be reduced little effect on the performance.

[Giuliani et al, 2011] presents the details of how they attempted to estimate the power consumption of data centre servers and storage. They decomposed the design process into multiple modelling phases and the best case error rate of the models evaluations was 2%.

[G. Contreras and M. Martonosi, 2005] predicts server and memory power consumption by using performance counters and events in the processor core. They apply a linear regression method to leverage the models.

[Rivoire, 2008] compares the full system power models based on resource utilization. Their focus is on models enabling specific energy efficiency optimization on specific machines, from laptops to servers. Their model evaluation using wide variety of benchmark machines present some useful information on machine whose dynamic power consumption is not dominated by the CPU and also the ones with built in power-managed CPUs.

[Fan et. al, 2007] modelled the power usage by analysing the CPU utilization in a large-scale server cluster. They studied the characteristics of up to 15 thousand servers and presents the results. By using their modelling framework to estimate the potential of power management schemes, they conclude that even in nine-tuned application, there is a considerable gap (up to 16%) between achieved and triracial peak power usage.

[Heath et al., 2005] built a heterogeneous server cluster to measure the power consumption using request distribution analytical models they developed. They have also modelled resource utilization using the time it takes to send a request to another node to determine maximum throughput and power.

[Kadayif at al., 2001] used software utility-based virtual energy counters to estimate the energy consumption of equipment that could even be used at a very low level like registers, memory, and address buses.

[Joseph & Martonosi, 2001] proposed usage of hardware performance counters as proxies to estimate the energy consumption of processors. They also employed a sampling-based approach to measuring signal transition activity within the processors, which is a good indicator of energy consumption.

[Ranganathan, 2006] generated customizable power utilization models on servers which can then be applied to others by using software simulators.

[T. Li, L.K. John, 2003] presents power models for the run-time operating system. They managed to achieve an error percentage within 1% of their cumulative OS-level energy consumption models.

[H. David, 2010] used a new approach to measure memory power consumption and utilized it to improve efficiency. The propose new power measuring and power limiting algorithms for main memory. Their evaluation shows that the algorithms are capable of achieving 40% less performance impact, compared to other baseline power limiting models.

Carbon emission is one of the biggest problems faced by humankind. It is a major contributor to many environmental issues including global warming.

The carbon footprint which is also called the carbon dioxide emissions coefficient is the measurement of greenhouse gas including CO₂ by an activity. The activity in question related to this research is the operation of a data centre. The carbon footprint of a data centre can be defined as the carbon emission of the electricity consumed by the data centre [Bouley, 2012]. The US Environmental protection Agency (EPA) has set a target of reducing the carbon footprint by 20% for government data centres. In Europe, all countries of European Union have agreed to reduce the carbon emission level of 1990 by 8% in 2012 [Bouley, 2012].

The location of a data centre plays a major role in the carbon footprint. As an example, a data centre located in an area where power is generated in more environmentally friendly ways would generate a smaller carbon footprint, compared to a similar size data centre located in an area where most of the power is generated using coal-fired stations. Iceland is a good example of the former case where most of the power comes from renewable energy sources.

2.6 Critical Analysis Summary

Based on our survey results, there has been a significant amount of research published on estimating the data centre energy consumption. However, most research focused on CPU counter-based approximation.

Main drawback of [G. Contreras and M. Martonosi, 2005] is that their models work for CPUs but are not accurate enough for predicting memory power consumption. That's because there being no direct relationship between CPU performance events and memory utilization. Also, the models are not abstract enough to be extended to other components beyond CPU and memory.

The drawback of [Fan et. al, 2007] approach is that it is more leaned toward the CPU utilization. Hence it did not take other server features like memory and age into account.

The assumption of all the components are linearly independent in [Rivoire, 2008] and restricts the generation of complex models. Further to that their calibration suit being based towards the idle case produces worst-case errors at the model evaluation using high utilization benchmark data.

The limitation of [Heath et al., 2005] is that it can only be applied to clustered servers, but most servers in data centres are not formed as clusters.

Due to heavy reliance on simulated OS level parameters, [T. Li, L.K. John, 2003] models cannot be applied to the broader range of equipment power consumption estimations.

The limitation of [H. David, 2010] is that it does not cover the other energy consumption aspects.

[Stanojevic and Shorten, 2010] evaluated the algorithms only on synthetic scenarios. It lacks the dynamic speed scaling algorithms testing using realistic traffic models.

2.7 Summary

What motivated us to carry out this particular PhD research is discussed at the top of this chapter. The research questions we have directed the research upon and how this thesis is organized is provided next.

We reviewed and presented the existing literature on estimation and prediction of energy consumption in data centres. Details on matrices used to benchmark energy efficiencies in DCs is also discussed with some example use case scenarios.

To address this area, we decided to direct our research towards equipment level modelling and machine relearning algorithms-based approximation by utilizing existing data from reliable sources.

Part II

Analysis and Modelling

“A man is a success if he gets up in the morning and gets to bed at night, and in between he does what he wants to do.”

-Bob Dylan

MODEL BASED ESTIMATION

3.1 Introduction

This chapter provides an overview of the basic concepts and comprehensive model analysis on estimating the energy consumption of each data centre equipment type.

3.2 Background on Energy Consumption

Energy consumption in data centres can be categorized into two stages.

1. **Initial setup:** First installation of the equipment and facility. This includes installing servers, air conditioners, and building racks
2. **Operation of the data centre:** Running the facility and IT resources

The focus of this research is only on the operational aspect.

Total energy consumption of the data centre includes the wastage during the power transmission from the generators to the point of use. Due to the difficulty of measuring losses during the transmission, this research focuses only on measuring power consumption within the data centre facility.

Measuring the total and equipment level energy consumption paves the ways for making informed decisions on energy efficiency.

As in equation 1, the total energy consumption is the summation of the total consumption of computation, Communication, and Infrastructure.

$$***E Total = E Computational +***
E Communication + E Infrastructure$$

Equation 1: Total DC Energy Consumption

Computation is mainly the servers with processing, memory and input/output handling. Communication is network devices which provide the connectivity between servers. Cooling systems like air conditioners and lighting in the premises fall into infrastructure.

Processors level power management needs to be taken into account in measuring computational power consumption. Different products employ different power management techniques and use multiple voltage domains for cores, cache and memory.

Data centres run “always-on” mode. Even when no processing taking place, all the equipment is still kept on active idle mode. This has both advantages and disadvantages. The biggest advantage of this strategy is relatively less overheads. It is because no application process, boot-up or warming-up is required as they are already up and running. These overheads can easily be overcome with a good strategic design model. The disadvantage is obviously the wastage of power to run the equipment when they are not being utilized.

When measuring the server energy consumption, both utilized and idle power should be taken into account.

Storage devices and main memory are the highest power consumers after processors.

Given that system components interact with each other, focusing on the individual unit may not be the best approach from the system standpoint.

3.3 Energy Efficiency Metrics

There are two major areas of data centre energy consumption. They are as below:

- 1. Information Technology (IT):** The energy used by IT equipment. Examples are servers, storage, and networking equipment.
- 2. Infrastructure:** The power consumed by the infrastructure equipment like air conditioners and lighting.

Several matrices have been proposed to identify the energy efficiency in data centre facilities. The one from the green grid is one of the most popular matrices.

3.3.1 The Green Grid Benchmarking Standards

The Green Grid (www.thegreengrid.org) is a leading trade association in Information Technology that consists of IT professionals. They are actively working on improving data centre energy efficiency. In line with this objective, they have proposed two major benchmarking standards of DC energy efficiency, which have been adopted by the broader research community and the industry. The two matrices are as below:

3.3.1.1 Power Usage Effectiveness (PUE)

PUE is an indicator of a data centre deficiency, introduced by the Green Grid [greengrid.org]

It is an excellent indication to see if a particular data centre needs any improvement in the process or the technology. Figure 5 shows how the total power coming into DC is consumed by IT and Non-IT equipment.

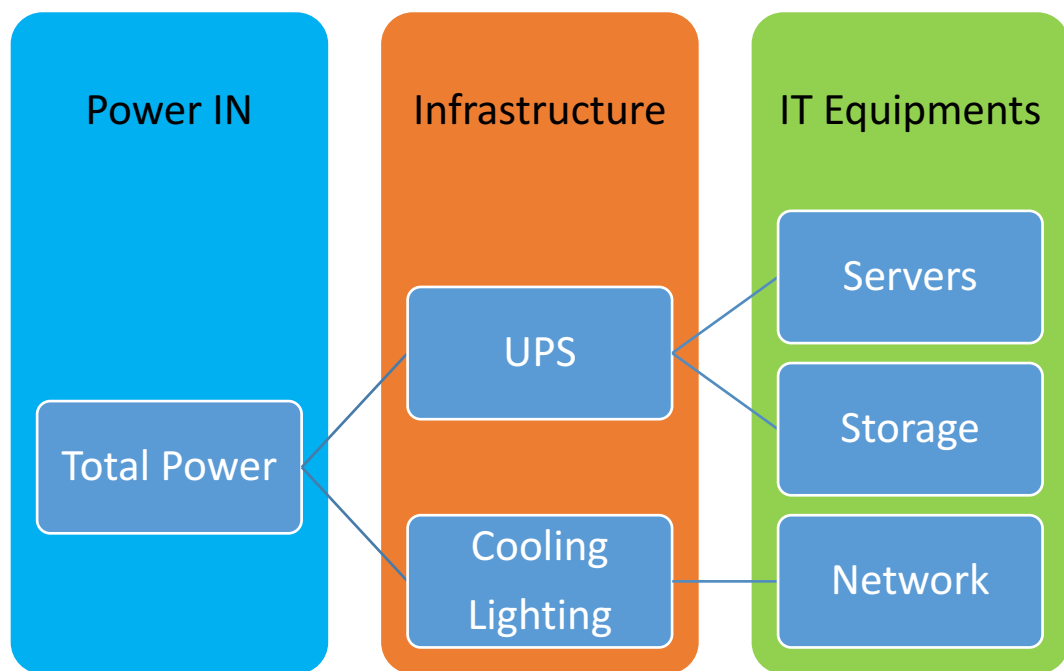


Figure 5:Energy usage in DC

So, the PUE is defined as in equation 2

$$PUE = \frac{\text{Total energy consumed by a data center}}{\text{Total energy consumed by IT equipment}}$$

Equation 2:PUE

The ideal but non-realistic PUE value is 1 which means all power in the facility is consumed by IT and non-infrastructure equipment like cooling fans, air conditioners, or light bulbs used in DC. Also, no energy loss at all in the transmission, conversion or consumption.

In a real-world scenario, a value around 2 is considered to be an average case for a data centre, meaning that about 50% of the energy used for non-IT equipment like coolers. A value between 1 and 1.5 is considered as efficient.

Case Study

Below is an example:

Let us assume the total energy used by the DC is 100MW and a breakdown of each aspect as in table 6.

	Equipment type	Example consumption (MW)	Total (MW)
IT	Servers	25	40
	Storage	10	
	Network devices	5	
Infrastructure	Air Conditioners	35	60
	Cooling Fans	10	
	UPS	10	
	Lights	5	

Table 6:Energy Consumption Example

These example figures are visualized in a pie chart in figure 6 below:

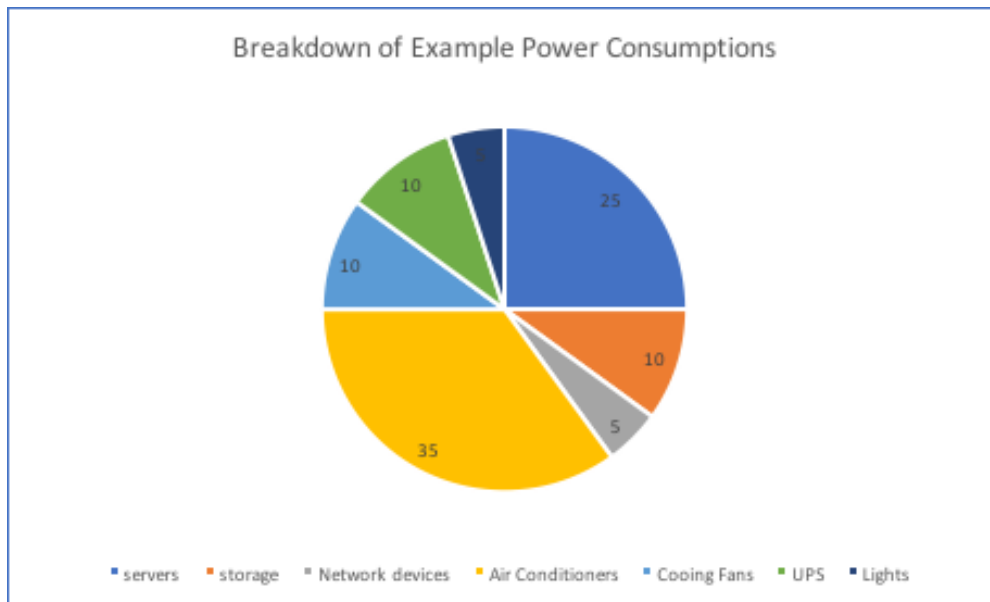


Figure 6: Breakdown of Consumption

applying the equation above, we get the PUE as below:

$$\begin{aligned}
 \text{PUE} &= \text{Total Energy} / \text{IT Energy} \\
 &= 100 / 40 \\
 &= 2.5
 \end{aligned}$$

Drawbacks of PUE

Even though PUE is an excellent matrix for benchmarking data centre efficiency, there are a few drawbacks. Major one is that some environmental factors can have an impact on PUE. As an example, in the summer months, PUE can be low compared to colder months as more energy is spent on air conditioners and other cooling equipment [P. Bemis, 2012]

3.3.1.2 Data Centre Infrastructure Efficiency (DCiE)

This is also introduced by the Green grid.

DCiE is the reciprocal of Power Usage Efficiency (PUE).

$$\boxed{\text{DCiE} = 1 / \text{PUE}}$$

Equation 3: DCiE

3.4 Power Consumption Analysis

Understanding the basic concepts around DC energy consumption is an integral part of model development. Details on each equipment category and associated variables of the total power consumption models are discussed in this section.

3.4.1 Categorization

We studied the energy consumption in data centres under the equipment categories listed under the subsections below. This is due to the differences in characteristics of each category and parameters associated with them. That helped us to come up with component level models which contribute to the total energy consumption.

3.4.1.1 Server

Servers provide the vital computation and processing power but not-surprisingly require a significant amount of energy. Servers consume power not only when they are utilized but when in idle mode as well. Idle power consumption can be considerably high as much as 50% of the fully utilization power [L. Barroso, U. Hölzle, 2007] and that is a major factor when analysing server power consumption. Due to the relatively high-power consumption during idle mode, it is important to keep the server utilization at a higher percentage to improve the efficiency of a data centre.

3.4.1.2 Storage

Data is stored on hard drives and they contain a number of disks inside. There are two common types of hard drives named - HDD (Hard Disk Drive) and SSD (Solid State Drive). SSD offers faster read/write operations and low latency but at the same time are more expensive compared to HDD.

The power consumption of a storage disk depends on the number of disks it contains.

3.4.1.3 Network

Network equipment provides the connectivity between computer devices, mainly the servers. This equipment (normally network switches) contains a number of ports. The energy consumption of a network switch is related to the number of ports it has. Compared to other equipment, network devices utilize less energy.

3.4.1.4 UPS

A UPS is a piece of infrastructure equipment which helps to provide regulated power supply. Unlike other types of devices, a UPS is designed and built to run at a higher level of utilization and are hardly switched off as the data centre needs to be powered all the time.

3.4.1.5 Cooling

The server room generates huge amount of heat. Equipment like air conditioners and cooling fans are used to keep the server room temperature down.

Cooling is usually the biggest power consuming aspect. Servers and storage come next. The next sections provide analysis details on the consumption of each equipment type.

3.5 Variable Definition

We defined the main variables which are identical to each piece of equipment They are listed as below:

- **Idle Power consumption**

Most IT equipment is not switched off or hibernated while not being utilized. They are ready to serve the next request as soon as it arrives. Hence, they consume power even in this idle status.

- **Fully Utilized Power consumption**

Equipment that consumes the most power when it is fully loaded. The technical term for this status is “fully utilized”. This variable defines the energy consumption when particular equipment is under 100% utilization.

- **Utilization percentage**

How much of average percentage a piece of equipment is in non-idle mode. The range of values is between 0 and 1. As an example, this value for a UPS is 1 or very close to that.

- **The efficiency of the equipment**

No equipment uses the full power it consumes in the main operation. There is always wastage and a common example is in the form of generating heat. This variable defines the percentage of power used for the actual utilization of equipment (inverse of the wastage).

- **Country of the data centre**

The country where a particular data centre is located. That plays a significant role as the cost of electricity is different from one country to another.

Also depending on the geographical location of the country, energy consumption for cooling may vary. As an example, less energy is required on cooling in a colder country such as Iceland.

- **Cost of electricity unit of the country**

Cost of electricity varies from country to country. This variable is for the country where the data centre is located.

3.6 Models

Using the variables as described above, we came up with energy consumption model for each category as discussed in detail below:

3.6.1 Server

In a data centre, the servers belong to different server types and specifications.

To work out the total energy consumption of servers, for each server type we need to calculate two components and add them together. The first component is the product of fully utilized power and the utilization percentage. The second, is the product of idle power consumption and the idle percentage (1- utilization percentage). Adding up both components gives us the total energy consumption of one server of that particular server type. Multiplying that number by the number of servers of that type is the total energy consumption of that server type.

The summation (Σ) of the consumption of individual server type is the total energy consumption of all servers in the data centre. The model is shown below:

$$E(\text{Energy}) = \sum_{n=1}^N (E_f u^n + E_i(1 - u^n)) t^n$$

Equation 4: Total Server Energy

The descriptions of the parameters are in the below table:

Parameter	Description
N	number of server types
n	given server type
t	number of servers of each type
t^n	number of servers in type n
E_f	energy consumption when fully utilised. This can be taken from performance benchmarking sources like spec.org
E_i	power consumption when idle. This can also be found in the benchmarking sources.
u^n	utilization percentage of server type n. Values are between 0 and 1. This can be measured for time intervals and calculate the average.

3.6.2 Storage

The storage equipment has a power rating associated with it. The same is for the servers. The power consumption for each storage type is first calculated, and then they are all added up. Storage has a number of drives and each drive has an individual power rate.

The power consumption of a storage unit is the total of all its drives. The model below gives total Energy of the storage devices:

$$E(\text{storage}) = \sum_{n=1}^N P^n u^n d^n t^n$$

Equation 5 : Total Storage Energy

Parameter descriptions as below:

Parameter	Description
N	number of storage types
n	given storage type
P^n	power rating of a given drive type
u^n	average utilization percentage of the drive type n
d^n	number of such drives
t^n	number of storage units of type n

3.6.3 Network

Network units also come with a power rating. These can be found in the manufacture's manual. The total energy of networking is given by the model below:

$$E(\text{network}) = \sum_{n=1}^N P^n u^n t^n$$

Equation 6 : Total Network Energy

Parameter descriptions as below:

Parameter	Description
N	number of network types
n	given network type
p^n	Power rating of a given network type
u^n	utilization percentage of network type n
t^n	number of network units of type n

3.6.4 UPS

A UPS is used in the data centres to provide uninterrupted power supply, in the event of an outage of the mains power supply. They are always associated with efficiency as a considerable amount of power is lost in the conversion process inside. So, in addition to other parameters, the power consumption model has to take efficiency into account as well. The below model gives the total energy of a UPS:

$$E(UPS) = \sum_{n=1}^N \left(\frac{P^n u^n}{e^n} - P^n u^n \right) t^n$$

Equation 7: Total UPS Energy

Parameter descriptions as below:

Parameter	Description
N	number of ups device types
n	given UPS type
p^n	power rating of a given ups device type
u^n	utilization percentage of ups device type n
t^n	number of CPU devices of type n
e^n	efficiency of the UPS type n

3.6.5 Cooling

The coolers may not always be used. As an example, on a very cold day, the cooling unit may not be turned off during the entire day. Due to that, the utilization percentage plays an important role in the model. Total energy of cooling is given by the below model:

$$E(\text{cooling}) = \sum_{n=1}^N P^n u^n t^n$$

Equation 8 : Total Cooling Energy

Parameter descriptions as below:

Parameter	Description
N	number of cooling equipment types
n	given cooling unit type
p^n	power rating of a given cooling equipment type
u^n	utilization percentage of cooling type n
t^n	number of cooling units of type n

3.4.6 Total Power Consumption

Total energy consumption for the whole data centre can then be represented as below:

$$E_{Total} = E_{server} + E_{Storage} + E_{Network} + E_{UPS} + E_{Cooling}$$

Equation 9: Total Energy of all equipment

3.7 Example

Below is an example which uses the result of surveying a small sized data centre. The source of the data is the Eureka project. Eureka project (www.dceureca.eu) is a research group founded to help European public sector data centres stakeholders to improve the digital platform procurement process. The help is available in the form of training, consultation and knowledge sharing. The primary objective of the project is to make the data centres more environment friendly by increasing the efficiency.

All equipment in the DC is listed with the quantity and then grouped into each category. The models from the previous sections were executed using the surveyed data. Individual data for each category and the calculated results are shown below:

Server Energy

Quantity	Equipment	Power Rating		Utilisation (0-1)
		100% utilised	idle	
3	Cisco UCSC210 M2	301	125	0.2
4	HP Proliant DL360	258	172	0.4
2	Dell PowerEdge 2850	258	172	0.4

$$\text{Server Power Consumption} = \sum_{n=1}^N (E_f u^n + E_i (1 - u^n)) t^n$$

$$= ((301 * 0.2) + 125 * (1-0.2)) * 3 + ((258 * 0.4 + 172 * (1-0.4)) * 4 + ((258 * 0.4 + 172 * (1-0.4)) * 2$$

$$= 1719 \text{ watts}$$

Storage Energy

Quantity	Equipment	Drive rating (Watts)	Number of drives	Utilisation (0-1)
1	EMC VNXe3300	450	1	0.8
1	Quantum Scalar i40 Tape Library	180	1	1

$$\text{Storage Power Consumption} = \sum_{n=1}^N P^n u^n d^n t^n$$

$$= 450 * 0.8 * 1 * 1 + 180 * 1 * 1 * 1$$

$$= \mathbf{675 \text{ watts}}$$

Network Energy

Quantity	Port OR Net Equip	Power Rating (Watts)	Utilisation (0-1)
2	Cisco C2960G iSCSISwitches	120	1

$$\text{Network Power Consumption} = \sum_{n=1}^N P^n u^n t^n$$

$$= 120 * 1 * 2$$

$$= \mathbf{240 \text{ watts}}$$

Cooling Energy

Quantity	Equipment	Power Rating (Watts)	Utilisation (0-1)
2	TOSHIBA Air Conditioner RAV-SM566KRT-E	5500	1
1	Daikin Inverter - FTKS60BVMB	5500	1

Applying the calculation model,

$$\begin{aligned}
 \text{Cooling Power Consumption} &= \sum_{n=1}^N P^n u^n \\
 &= 5500 * 1 * 2 + 5500 * 1 * 1 \\
 &= \mathbf{1650 \text{ watts}}
 \end{aligned}$$

UPS Energy

Quantity	Equipment	UPS Rating (Watts)	Utilisation (0-1)	UPS Efficiency
2	APC Smart-UPS 3000	2700	0.5	0.93
1	APC Smart-UPS 6000	4200	0.5	0.93
2	APC Smart-UPS 2200	1920	0.5	0.93
2	APC Smart-UPS 1000	700	0.5	0.93

Applying the calculation model,

$$\text{UPS Energy consumption} = \sum_{n=1}^N \left(\frac{P^n u^n}{e^n} - P^n u^n \right) t^n$$

$$\begin{aligned}
&= (2700 * 0.5 / 0.93) - 2700 * 0.5 * 2 \\
&\quad + (4200 * 0.5 / 0.93) - 4200 * 0.5 * 1 \\
&\quad + (1920 * 0.5 / 0.93) - 1920 * 0.5 * 1 \\
&\quad + (700 * 0.5 / 0.93) - 700 * 0.5 * 1
\end{aligned}$$

= **558.39 watts**

Total Consumption of the DC

Total energy consumption = energy consumption of (servers + storage network + UPS + Cooling) +

$$= 1719 + 675 + 240 + 1650 + 558.39 \text{ watts}$$

= **4842.39 watts**

3.8 Energy Snapshots

The lack of ability to see the impact of potential improvements to the equipment portfolio is a big challenge faced by DC managers. To make effective decisions and justify the cost associated with them, they need to be able to preview the impact of such improvements. Previewing the impacts of the changes would help them to identify opportunities for energy savings and areas for improvement.

To fill that gap, we propose the concept of *Energy Snapshots*. These snapshots help them to compare between different states of their portfolio by making virtual changes to them.

There are two types of snapshots as below:

Baseline – The initial energy profile of the data centre. That takes all the parameter values for each equipment type and feeds the models to calculate the total energy consumption. These initial snapshots are used as the baseline to compare against future improvements.

Comparison – A new image is constructed after a change of parameter values and their associated total consumption by executing the models. The profile is in fact a collection of data model objects which reflect the current state of the data centre equipment portfolio.

Figure 7 depicts how Comparison and Baseline snapshots are organized to provide useful insights into potential energy efficiency improvements.

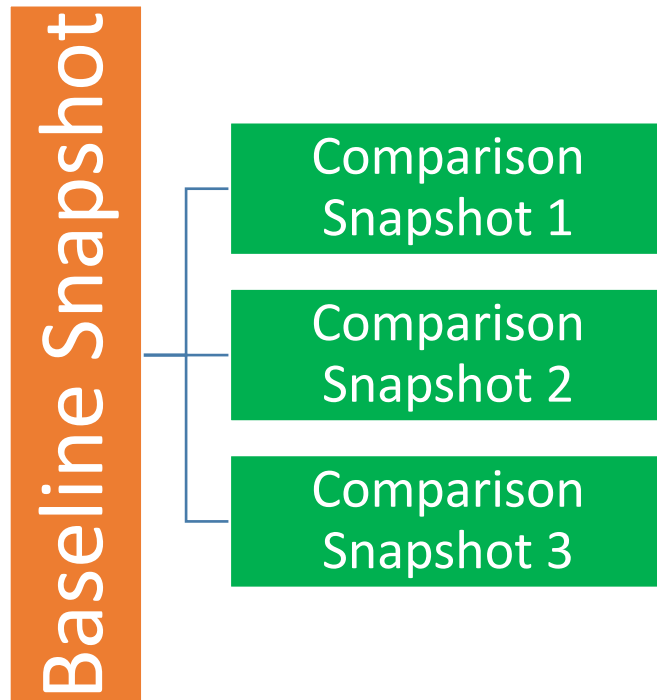


Figure 7:Snapshot organization

Details on how these energy snapshots were implemented and calculations carried out are discussed in chapter 6. Evaluation details using real-world data is also presented there.

The tool which helps to compare the snapshots provides the DC stakeholders with a powerful platform to benchmark their current energy consumption and the real impact of changes. That also provides an opportunity to decide on the next step and helps to define goals for improvements.

Figure 8 below summarises how the energy snapshots can be used to plan and evaluate a potential improvement.

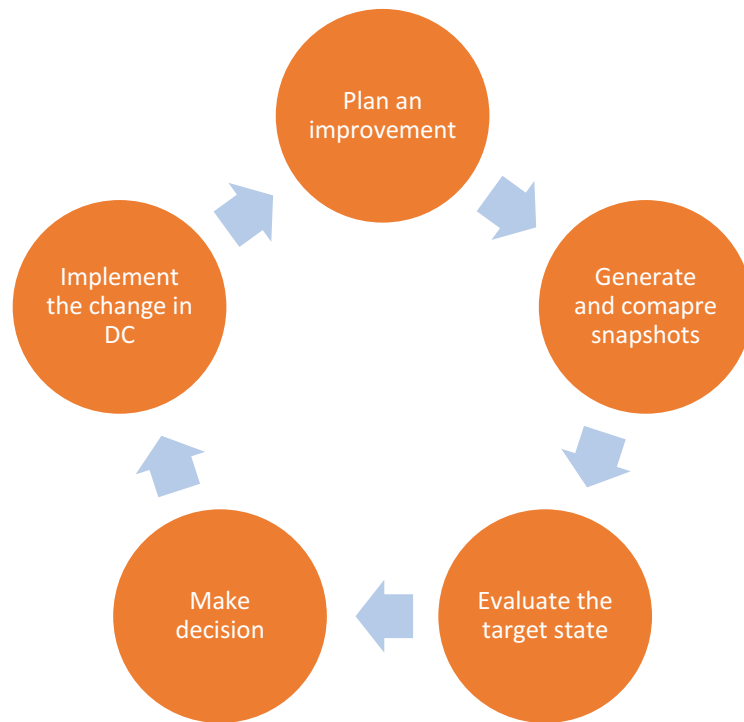


Figure 8:Workflow of Snapshots

3.9 Conclusion

We discussed how the calculation models are used to estimate energy consumption. Firstly, by identifying the common and quantifiable attributes of the data centre equipment. Then we presented analysis on how those individual variables factor into the total energy consumption of each piece of equipment. As a result, unique models and equations were derived. We discussed how to estimate the energy consumption using those models and equations without having to physically measure them. This is one of the major research questions we undertook.

We then proposed the snapshot concept and how it can be used to plan and compare the change of a potential improvement.

How these models were implemented by a tool and evaluated using test data are discussed in chapter 6.

MALEP v1: LINEAR REGRESSION PREDICTIONS

4.1 Introduction

Why machine learning is considered and chosen in this research is the initial discussion of this chapter. The introduction of related ML techniques and why they are relevant to this particular research is also discussed. The comprehensive analysis to find suitable models and details on how machine learning was leveraged to predict energy consumption in data centres is also presented with research evidences.

Below are the areas which we led our research to predict the energy consumption using ML. Those come under main research questions mentioned in chapter 1.

- What is the available ML offering?
- How to select the best ML method?
- How to choose training data set for the ML model?
- How to validate prediction using test input data?

4.2 Why Machine Learning in this research

We implemented the models and equations presented in chapter 3 in a highly user-interactive tool. The tool enables the stakeholders of a data centre to set up the overall energy consumption profile by providing known input details. The tool is designed in such a way that it offers alternative options if particular information is unknown or unavailable.

Once the parameter values are provided, the system executes the calculation models to estimate energy consumption. The estimated values, along with user input details, are stored in the system. When the user logs back into the system later on, they can amend the details and re-run the models to generate the new estimations.

4.2.1 Idle and fully utilized power consumption

Finding required parameter values for the parameters for all other equipment models is not difficult. It is because most of them can be found in the user manuals or on the manufacturer's website. However, this is not the case for servers.

Idle and fully utilized power consumption are such examples where it cannot be found in the manuals.

However, there are organizations that carry out comprehensive benchmark testing to calculate those figures on servers for specifications from well-known vendors. Standard Performance Specification Corporation (SPEC) is one such organization. They carry out rigorous benchmark testing and publish the results regularly on their website(spec.org) Hence that information is publicly available.

But there is real challenge of using that data. Such published data is for known specifications. However, most of the server specifications found in the data centres are custom built ones. Hence the power consumption for such specifications are not available on the publish data from SPEC.

To overcome this problem, we decided that we need to estimate or predict the idle and fully utilized power consumption for the DC servers. Given that data is available for common specifications, the decision was made to go for data analytical prediction.

After careful consideration, we chose machine learning as the prediction technique.

4.3 Related Machine Learning Techniques

Machine learning is a subcategory of artificial intelligence which addresses the question of how to build computers that improve automatically through experience. ML is one of today's most rapidly growing technical fields. It is at the intersection of computer science and the statistics and the core of artificial intelligence and data science [Jordan & Mitchell, 2015]

Although ML appears to be a modern and more recent development, it has been a popular research area since the 1970s. This academic topic has picked up the pace recently, thanks to the rapid increase in the memory and processing power of the computers.

The high availability of big data in all scientific and statistical fields has also heavily contributed to its recent popularity.

Even though ML is usually considered as a computer science area, it is a multidisciplinary field. Fields like mathematics and statistics are equally applied in ML model development, training and evaluation.

Figure 9 depicts the essential elements and the workflow of Machine Learning.

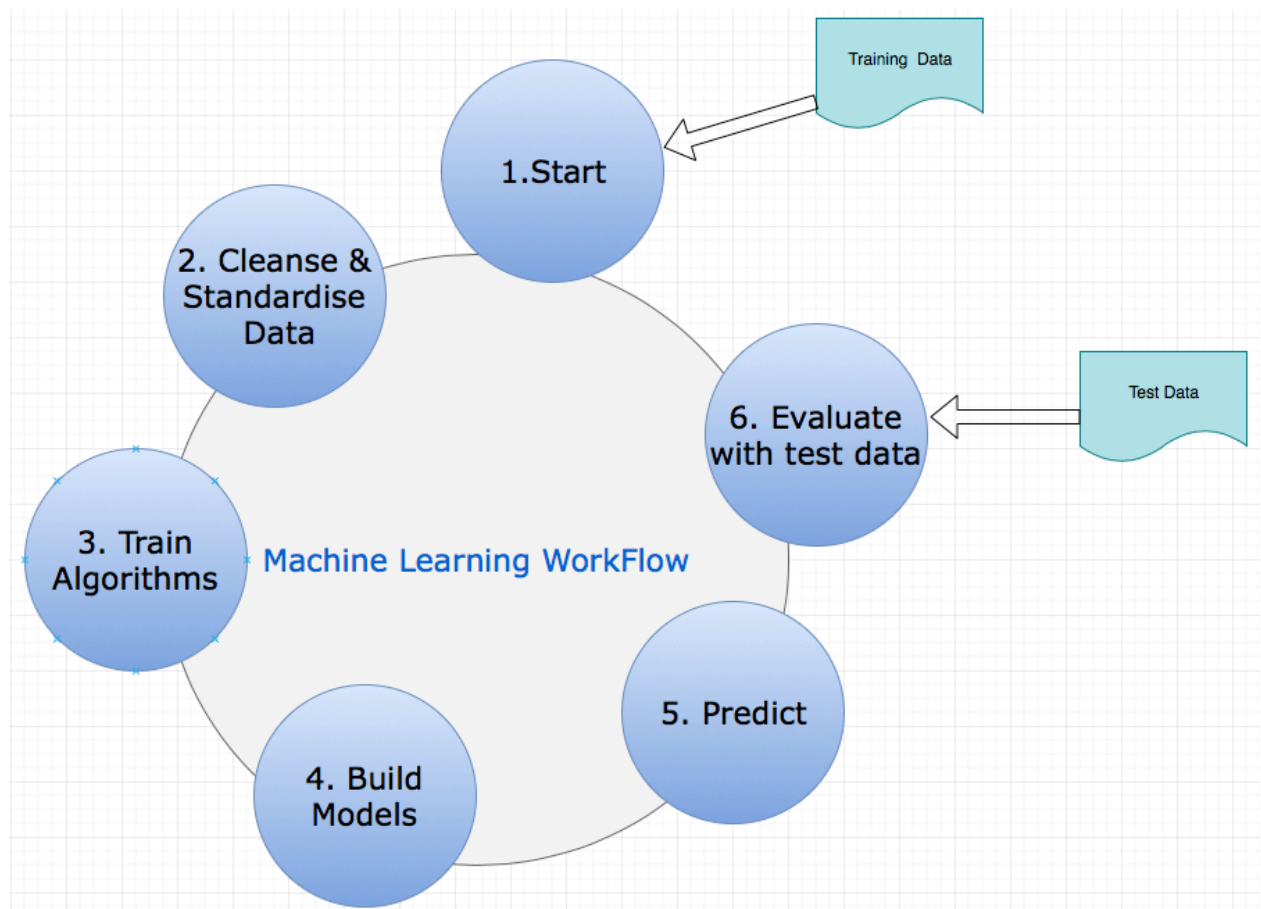


Figure 9:Machine Learning Workflow

At the heart of the process is the training data which is required to train the models. Most of the time, some sort of data cleansing is required to standardise this training data to leave out the anomalies and align to a suitable format.

Depending on the nature of the training data, a suitable ML algorithm needs to be selected. More details on the ML algorithms are discussed later. Then the models are constructed on the back of training data, based on the chosen algorithm.

The next phase is the prediction where the trained models are applied on new input data which are not part of the training dataset, to predict the likeliest outcome. Finally, the predicted outcome is evaluated.

The whole cycle can be applied multiple times for better results.

Machine Learning is subcategorized into two major areas, namely supervised & unsupervised learning. Figure 10 is a representation of these categorizations in the context of this research.

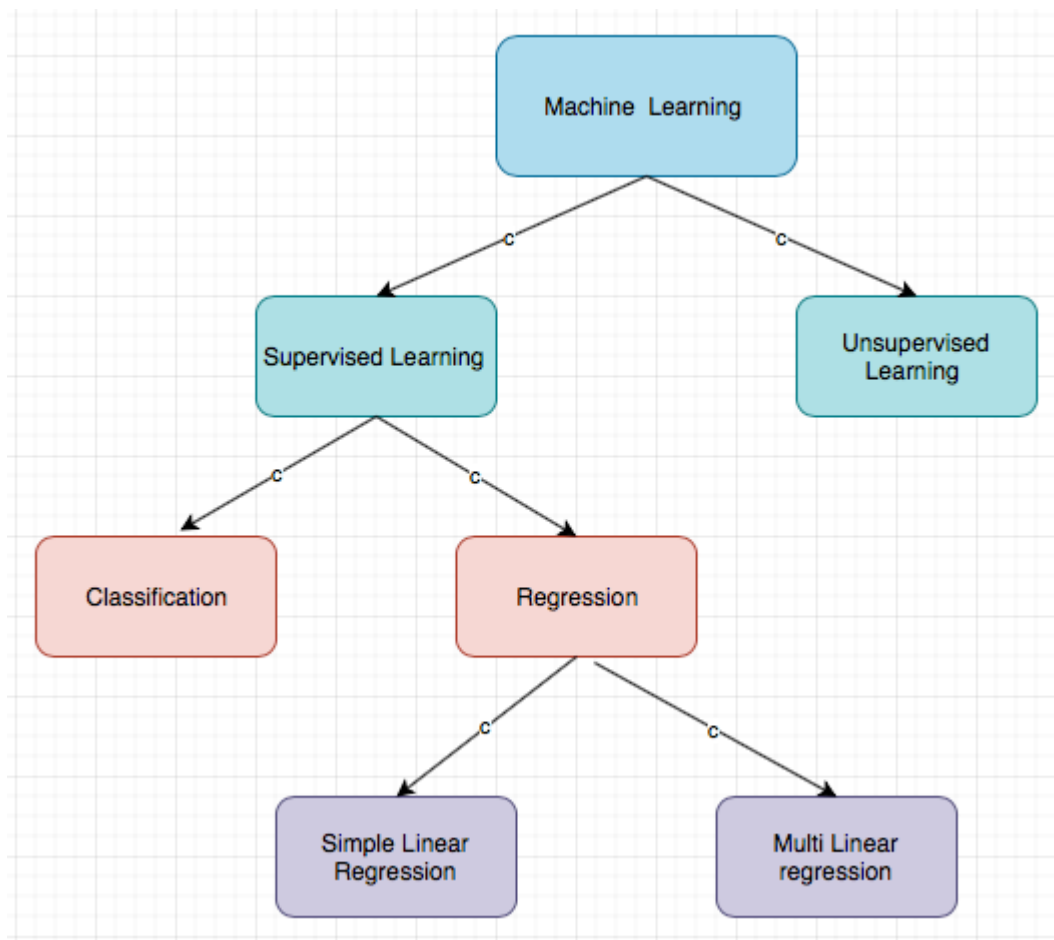


Figure 10:ML categorization

4.3.1 Unsupervised Learning

Unsupervised learning is the process of analysing a data set where the predicted outcome is unknown. Some deep learning algorithms are usually employed to investigate a data set to come up with previously unknown patterns.

“Unsupervised learning studies how systems can learn to represent particular input patterns in a way that reflects the statistical structure of the overall collection of input patterns. By contrast with SUPERVISED LEARNING or REINFORCEMENT LEARNING, there are no explicit target outputs or environmental evaluations associated with each input; rather the unsupervised learner brings to bear prior biases as to what aspects of the structure of the input should be captured in the output.” [Dayan, 1999]

Compared to supervised learning, unsupervised learning is more complex and challenging. This is due to the fact that there are no predefined labels or “targets”. Example applications for unsupervised learning are fraud detection in transaction banking, finding hidden patterns in supermarket sales and targeted marketing.

Clustering is the most common methodology here. In clustering, learning algorithms are applied to the data to find out characteristics and hidden patterns. Example algorithms commonly used are K-means and probabilistic clustering.

Association is another methodology in unsupervised learning. Association tries to discover rule sets that relate one data set into another. One example is to discover purchasing patterns in supermarkets. For example, the different products usually bought by the customers who purchase hair products while shopping.

4.3.2 Supervised Learning

Supervised learning is the process where the predicted outcome is a set of known values or results. It is the computational task of learning correlations between variables in annotated data (the training set), and using this information to create a predictive model which is capable of inferring annotations for new data, whose annotations are not known [Fabris & Magalhães , 2017].

Email and SMS spam [Abdulhamid, 2017] filters are some examples where the predicted outcomes are spam or not-spam. Diagnosing diseases using medical data [Dasgupta, 2011] is another example where the outcomes are positive or negative.

Supervised ML is performed under two different categories called Classification & Regression which are discussed in detail later in this chapter.

4.3.2.1 Classification Problems

Classification problems are where the predicted outcome is a discrete classified set (labels). The classification can be binary or multi-class. In binary class classification, the possible number of outcomes is two. Classifying whether male or female by analysing features, deciding if infected vs not infected with certain diseases using medical data are examples for binary classification.

In multi-class classification problems, the number of predicted outcomes is more than two. Some of the examples are predicting the colour of marbles (red, blue, green) using the same feature set, categorising the class of apples (A, B, C) by analysing the features and recognition of handwritten letters.

Some of the algorithms used in classification are Linear Classifiers, Logistic Regression, Decision Trees, Random Forest, Naive Bayes, Nearest Neighbour and Neural networks.

As examples, [Osisanwo, 2017] compares the accuracy of some classification algorithms using a large data set.

[Kumar & Yadav, 2018] has details on the implementation of some classification algorithms in Python, considering a few practical examples.

4.4 Linear Regression

Regression problems are where the predicted outcome is a continuous series of values. House price prediction using features of properties or prediction of employee salary by using some of their attributes are some examples.

The fundamental of Linear regression is that the assumption of the linear relationship between the independent variables and the predicted dependent variable. The process observes continuous features to see they have a continuous relationship to the predictor. There can be one or more dependent variables which are also called regressors and they are denoted by X.

4.4.1 Coefficient

Coefficient, which is also called as Correlation is a numeric value ranges between -1 and 1. If the value is 0, there is no linear relationship between X and Y. The value of -1 or 1 indicates a perfect negative or positive association. The closer to the value of -1 or 1, the stronger the linear relationship between the independent variable (X) and the dependent variable (Y).

4.4.2 Simple Linear Regression

SLR deals with only two variables — the independent variable, which is represented by X and the dependent variable, which is represented by Y.

Y can be positively or negatively dependent on X and the relationship is a linear one. This relationship can be mathematically described in equation 10 as below:

$$Y = mX + b$$

Equation 10: Simple Linear Regression

The parameters are described in below table:

Parameter	Description
X	Independent variable
Y	Dependent variable
m	Coefficient
b	Intercept

Predicting the salary of an employee based on the age is an example application of SLR that is based on the assumption that in some industries, the older the employee is, the higher the pay.

4.4.3 Multiple Linear Regression

As the name suggests, MLR deals with more than two variables. As it was evident in the previous section, the number of cores in the server CPU is a significant factor towards the energy consumption of a server.

So, in this approach, both CPU speed and the number of cores were used as input variables for predicting the energy consumption of unknown server specifications.

MLR can be mathematically represented as below in equation 11:

$$Y'_i = b_0 + b_1X_{1i} + b_2X_{2i}$$

Equation 11 : Multiple Linear Regression

The parameters are described in the below table:

Parameter	Description
Y'_i	Dependent variable (predicted)
X_i	Dependent variables
b_0	Intercept (constant)
$b_1, b_2 \dots b_n$	Coefficients (slopes)

If we extend the salary prediction example given in 4.4.2, in addition to the age, we can also include some more attributes of an employee to predict the salary. The level of education, the number of certificates or licences an employee holds, the number of years in the current employment are such examples features to take into account.

When all these attributes are used to check if a linear regression exists between those employment related features and the employee’s salary, this is an example for an application of MLR.

4.5 Server Attribute Analysis

4.5.1 Server Actual age Vs Reference age

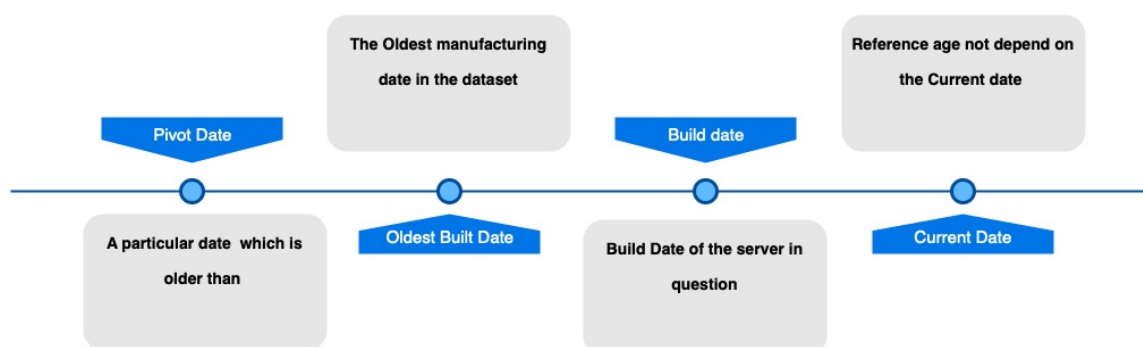
The actual age of the server in the data set is the difference in months between today and the month when the server was built. It can be an independent variable on its own, but there is a drawback of using it as a parameter. That is because today's date is a moving target and the age variable on the model performance will be less effective by the time.

To minimize that risk, we introduced a derived variable called “reference age” (in months).

It is calculated as below:

1. Find the oldest date (month and year) from the test data set (September 2004 in this case).
2. Use a pivot date point which is older than the date found in the step 1. The pivot date chosen here was January 2003.
3. The reference date is the difference between two dates.

$$\text{Reference age} = \text{server built date} - \text{pivot date point}$$



Below is an example showing calculated reference age by applying the above process for some selected servers. The chosen pivot date is 01 September 2004.

Built Date	Actual Age(In Months)	Pivot Date	Reference Age(In Months)
01/04/2008	134	01/09/2004	63
01/10/2008	128		69
01/09/2018	9		188
01/01/2019	5		192
01/05/2012	85		112
01/03/2010	111		86
01/09/2011	93		104
01/06/2010	108		89

4.5.2 Studied Attributes

The fully utilized and idle power consumption is the predicted values of the models and the values in the training data is used to train the models. All others are independent variables on their own or combined with other attributes, depending on the model. The test data set from standard performance evaluation corporation (spec) contains various other attributes like Java virtual machine vendor, but they were not selected as the likelihood of their impact on the power consumption very less. Models were executed and results were analysed before finalising the selected list of features.

To analyse the models for prediction of server energy, the independent attributes we studied are listed below:

Attribute	Measuring Unit	Type	Comment	Reason for Selection
Processor Speed	Megahertz (MHz)	Independent	Not difficult for DC operator to find out the value in real life.	CPU is a high-power consuming part of a server. The speed of the processor is a major contributor towards the performance of the CPU.
Total Memory	Giga Bytes (GB)	Independent	Not difficult for DC operator to find out the value in real life.	With a higher memory, more processing can be supported, hence total memory contributes the power consumption of a server.
System Age	Months	Independent	Not difficult for DC operators to find out the value in real life.	General assumption is that newer servers are more energy efficient as they are built with modern technology. So, the age of a

				server system is an important factor of the energy consumption.
Reference Age	Months	Independent	Derived from actual system age to standard the feature values.	As opposed to the system age, the effectiveness of this derived parameter does not change by the time.
Number of Cores	Distinct numbers	Independent	Value can be found in the paperwork or in the system information.	Number of cores directly related to the performance and the power consumption of a server as each core draws power.
Threads per Core	Distinct numbers	Independent	can be found in the paperwork or in the system information.	It tells how many processing units can be executed at the same time within a processor core. This count is also directly related to the power consumption of a server.

Power Supply Rating	Distinct numbers	Independent	can be found in the paperwork.	Mandatory parameter for most of the power consumption models.
Fully utilized power consumption	Watts (w)	Dependent	Not easy to find. Benchmark test organization publish the values only for known specifications.	Mandatory parameter for most of the power consumption models.
Idle Power consumption	Watts (w)	Dependent	Not easy to find. Benchmark test organization publish the values only for known specifications.	Mandatory parameter for most of the power consumption models.

Table 7 : Test Data Attributes

4.5.3 Outliers Discovery and Removal

In a dataset, outliers are the extreme values which diverge so much from the overall pattern and directly influence the error rate.

The cause of outliers can be varying. They may be due to mistakes in calculations or the noise added while collecting data.

In any machine learning exercise, it is recommended to carry out analysis on the outliers. Based on the discovery, they can be removed to increase the accuracy of an algorithm.

When removing the outliers, a few factors need to be taken into consideration. Firstly, the percentage of data that is identified as outliers. This is important because due to any reason, if a high ratio of data needs to be removed resulting in not enough data being left, you may have to re-consider your algorithm.

Our chosen strategy was to treat data points as outliers when they are below or above a few times of standard deviation from the mean. After some statistical analysis, we found out the effective number of standard deviations to be 2. The strategy is explained in detail below:

1. For a given field, the mean and standard deviation are calculated.
2. Lower and higher extreme thresholds are calculated

Lower extreme threshold = **mean – (2 * standard deviation)**

Higher extreme threshold = **mean + (2 * standard deviation)**

3. Values below the lower threshold or above the higher threshold are removed.

Impact Analysis

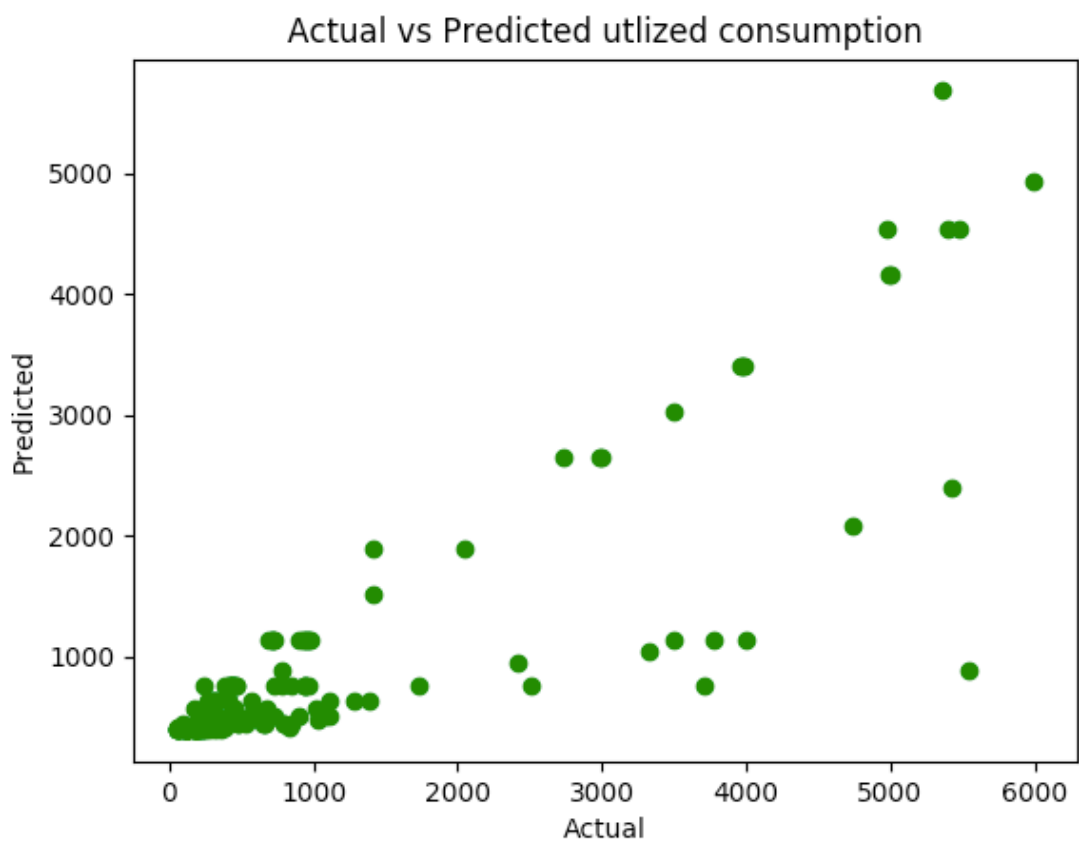
To put the outlier removing strategy into practice, we carried out a scenario as discussed below.

We considered predicting utilized power consumption using only memory as independent variable.

We ran the regression model and results are below:

Test results	Coefficient	Intercept	Average error %	Standard Deviation	Min Error	Max Error
224	1.97	379.59	96.69	120.56	2.82	762.03

A plotted graph is below:

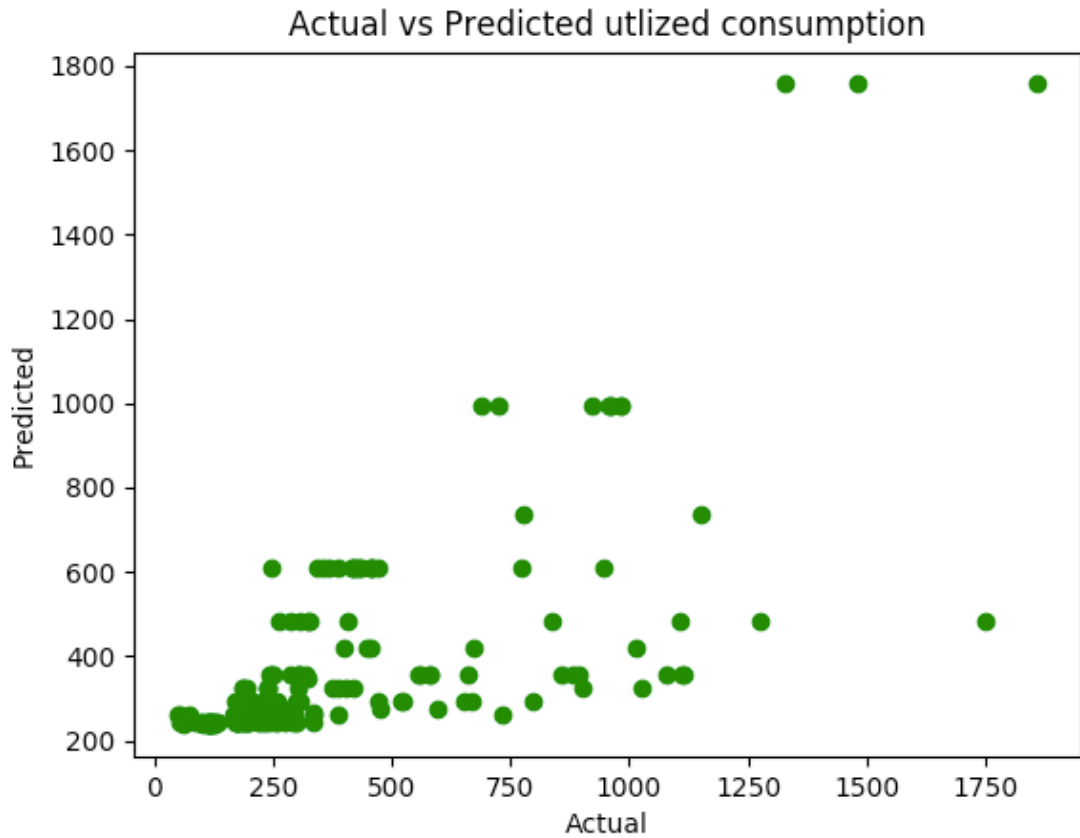


The average error is very high (96.69) and the maximum error as high as 762.03%.

We then re-ran the models without the outliers. Results are shown below:

Test results	Coefficient	Intercept	Average error	Standard Deviation	Min Error %	Max Error %
199	1.98	229.38	50.39	66.55	0.46	445.30

The same as before, the results are plotted in the graph below:



Comparing results, we see a reduction of 25 in test data rows. It has gone down from 224 to 199. Those 25 data items are the outliers based on the criteria we applied.

The comparison is summarized in the below table:

	With Outliers	Without Outliers
Test Data Set	224	199
Average Error Percentage	96.69%	50.39%.
Standard Deviation	120.56	66.55
Maximum Error Percentage	762.03	445.30

As a result, the error percentage has significantly improved from 96.69% to 50.39%.

This improvement can be explained by looking at the changes in standard deviation and max error percentage. They went down from 120.56 to 66.55 and 762.03% to 445.30% respectively.

4.6 Single Linear Regression (SLR) Analysis

Initial predicted energy consumption using the input parameters listed in table 7 indicated a continuous series of values which suggested some form of linear regression relationship between some of the input parameters and the predicted outcome. Based on that it was decided to carry out research into detailed analysis on the linear regression models. This section discusses how the ML models were analysed in the context of the prediction of server energy consumption.

Firstly, we started with SLR. Two variables selected were the processor speed (in MHz) and the predicted power consumption. A specific dataset was chosen only by considering the servers with eight cores. The reason for that that filtering was to standardise the data set.

X = server CPU speed (MHz)

Y = predicted fully utilised energy consumption (in watts)

The linear regression model was run on the selected data set, and the results are presented in figure 11.

A linear relationship between the processor speed (x) and the energy consumption (y) can be seen here with an estimated coefficient of **0.16**.

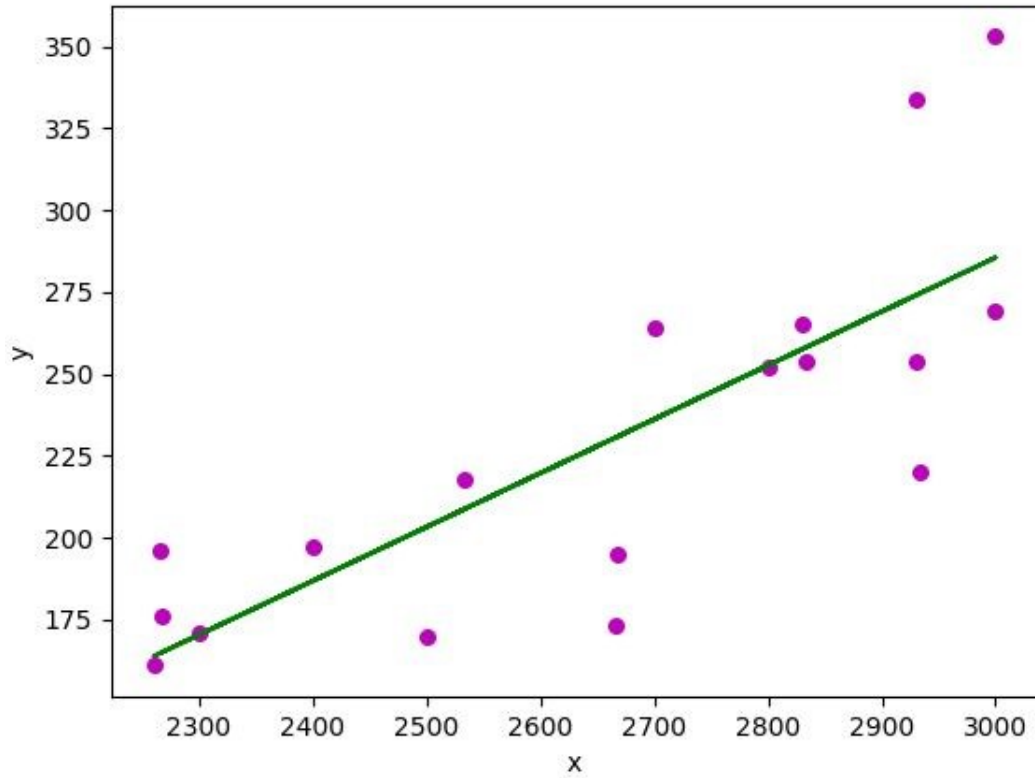


Figure 11:processor speed vs predicted energy consumption for selected dataset

However, there is a significant limitation of this approach as only a selected subset of training data (servers with 8 CPU cores) was used here.

We then wanted to examine how the model will behave on the entire dataset with a wide range of server cores. So, the model was run on an unfiltered dataset.

The predicted results as below:

Average error %	Intercept	Coefficients
58.01	526.38	-0.08

A plotted graph of processor speed and the predicted energy consumption on the whole data set is shown in figure 12.

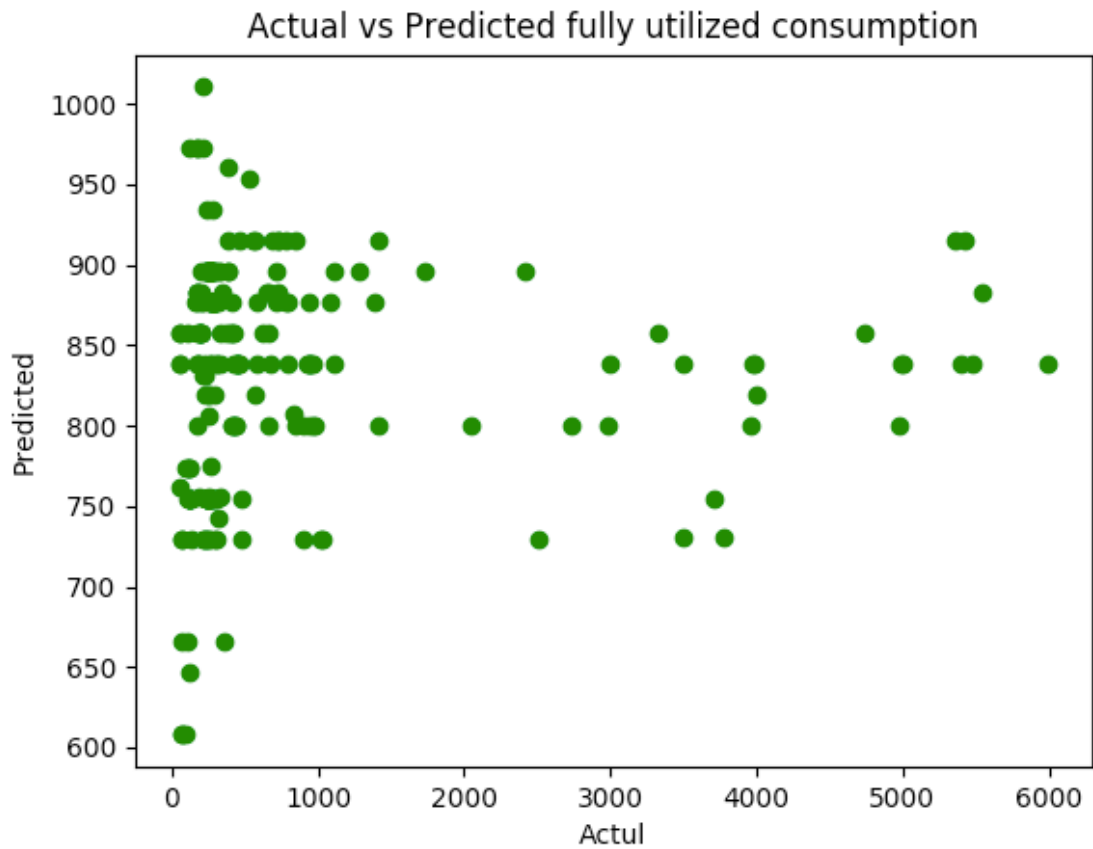


Figure 12: CPU speed vs energy consumption for the whole data set

Now as it can be easily seen, there is no good linear relationship between the processor speed and energy consumption anymore.

Shown below is the comparison between actual and predicted power consumption:

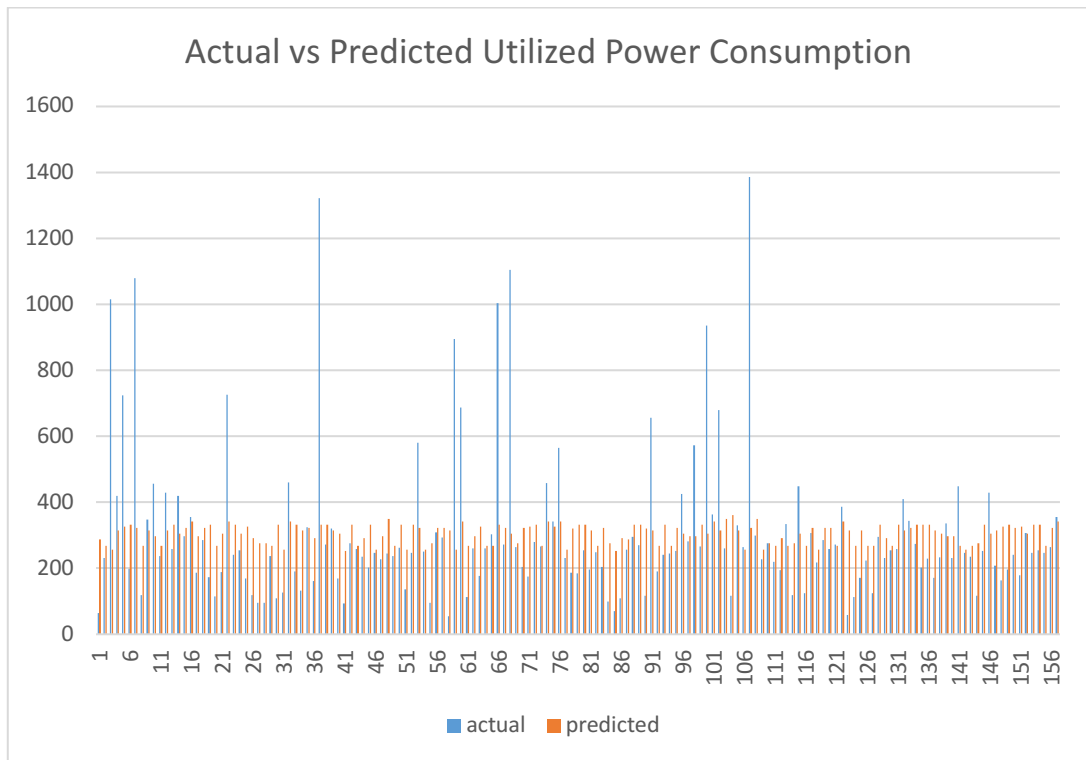


Figure 13:Actual Vs Predicted

SLR using number of cores

We then examined the relationship between fully utilized power and the number of cores of the processor. The predicted results are shown in the graph below:

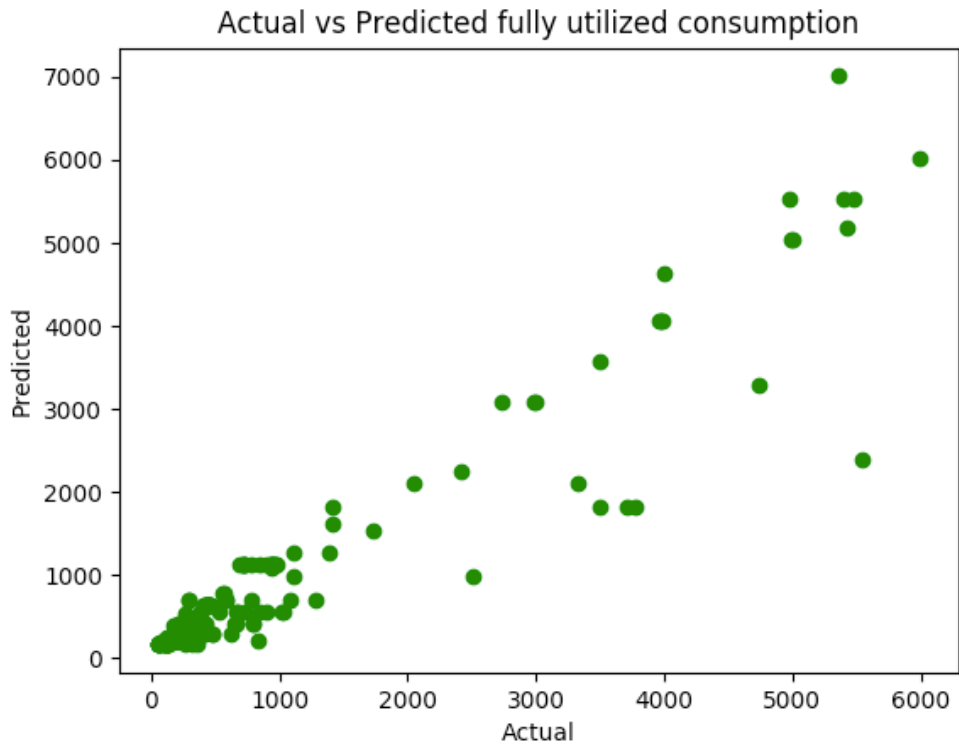


Figure 14:Actual Vs Predicted

The predicted results are below:

Average error %	Intercept	Coefficients
29.97	138.36	7.24

Based on the results above, the number of cores is a good candidate for predicting fully utilized power consumption. The Actual vs Predicted visualised below:

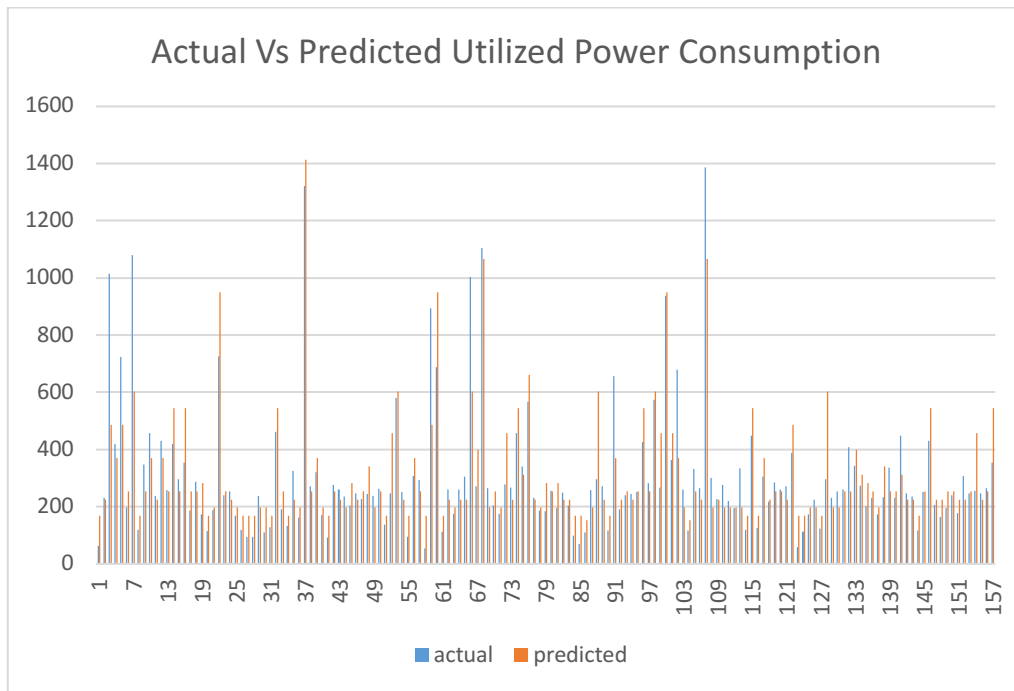


Figure 15:Actual Vs Predicted

However, the accuracy of the prediction of idle power consumption is significantly lower. As shown in the evidence below, the average error percentage is 55.48:

Average error %	Intercept	Coefficients
55.48	75.93	0.61

The plotted graph of actual vs predicted idle consumption for complete data set is shown below:

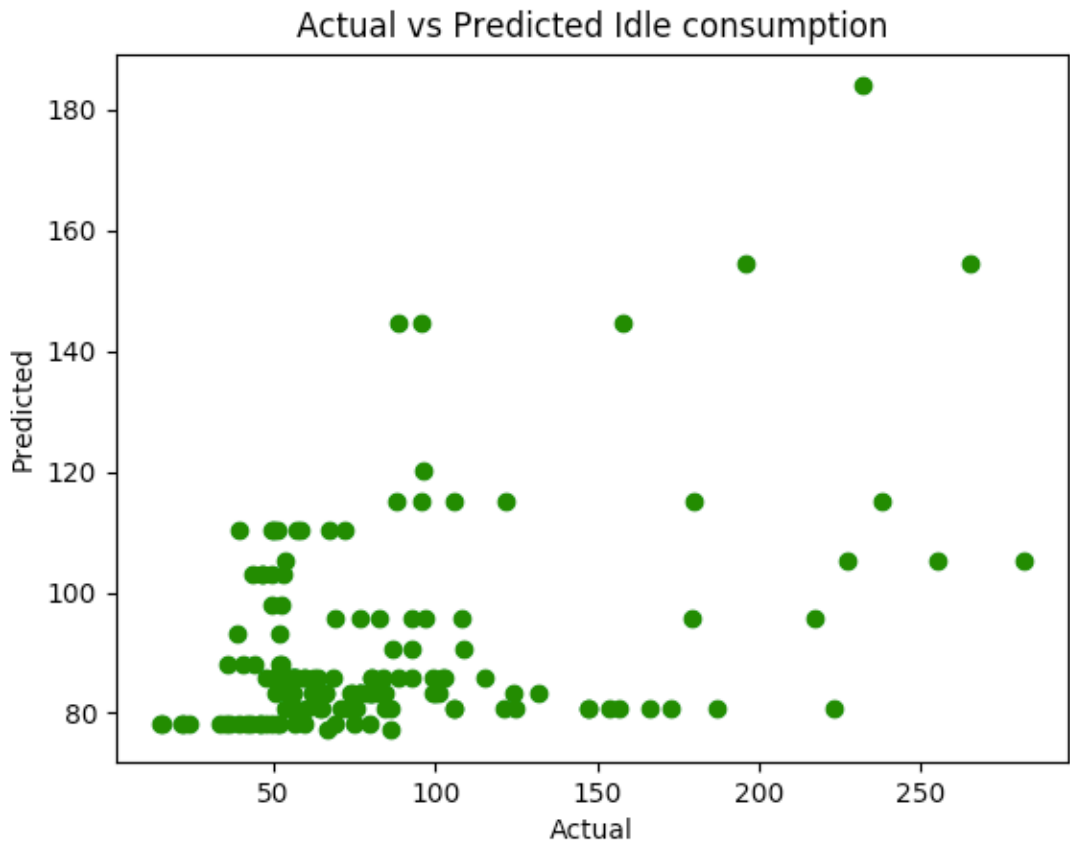
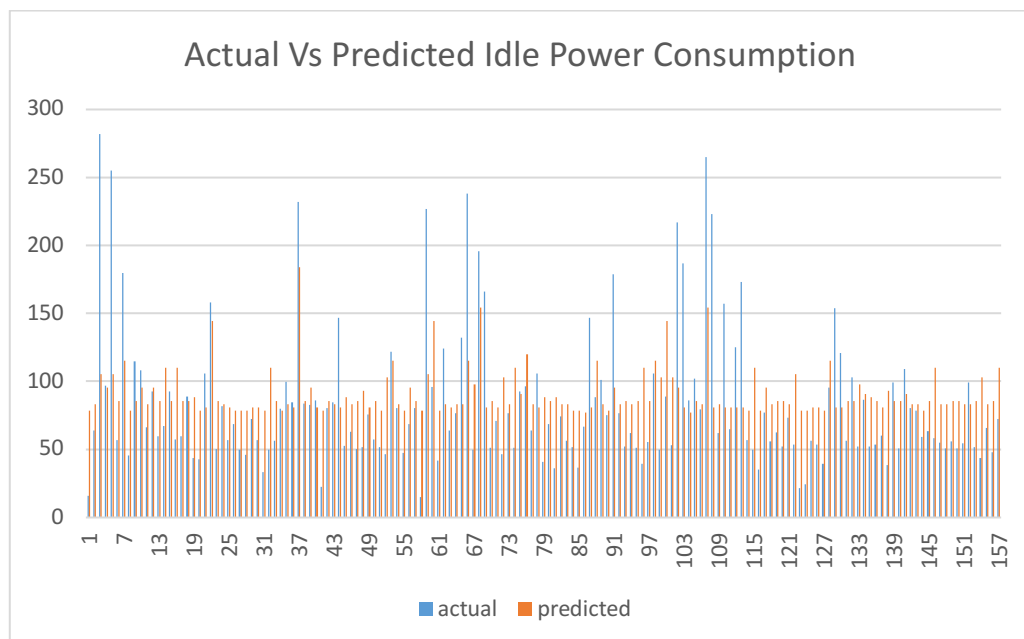


Figure 16:Actual Vs Predicted

And the Actual Vs Predicted is visualised below:



So, that indicates that number of cores is a less important factor of the energy consumption when in idle.

SLR using Memory

The results of when memory was taken as the independent variable is shown below:

Utilized Power

Average error %	Intercept	Coefficients
43.29	214.96	1.97

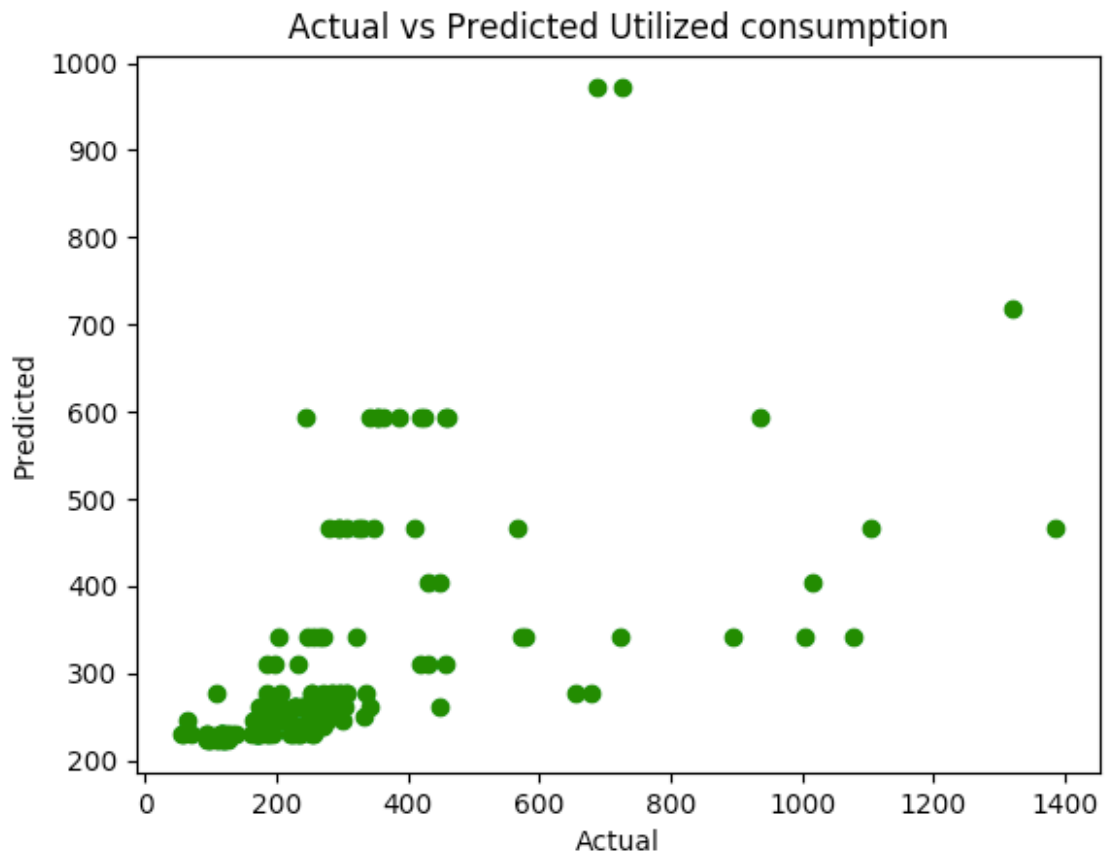


Figure 17: Actual Vs Predicted

Actual Vs Predicted in a line chart is shown below:

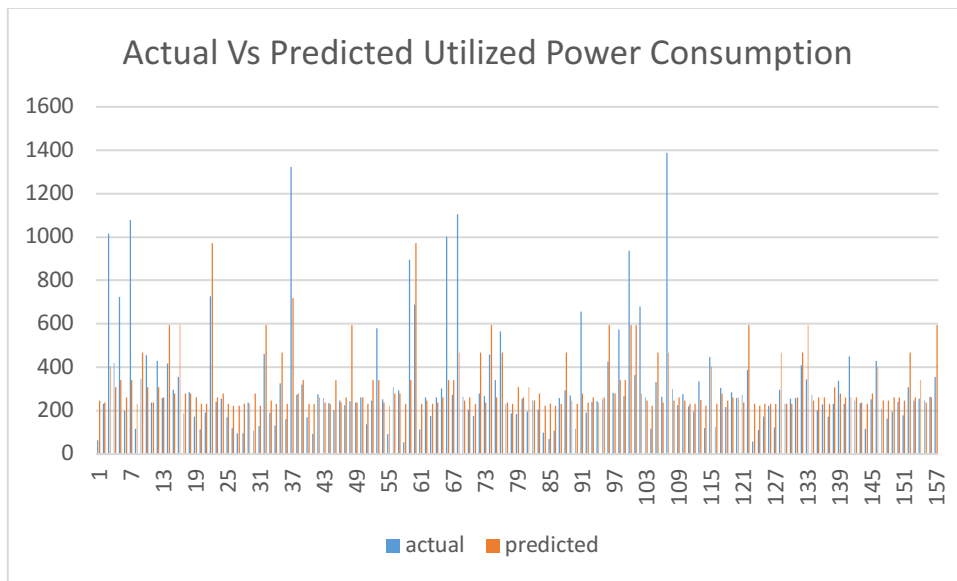


Figure 18:Actual Vs Predicted

Idle Power

Average error %	Intercept	Coefficients
59.01	86.92	0.06

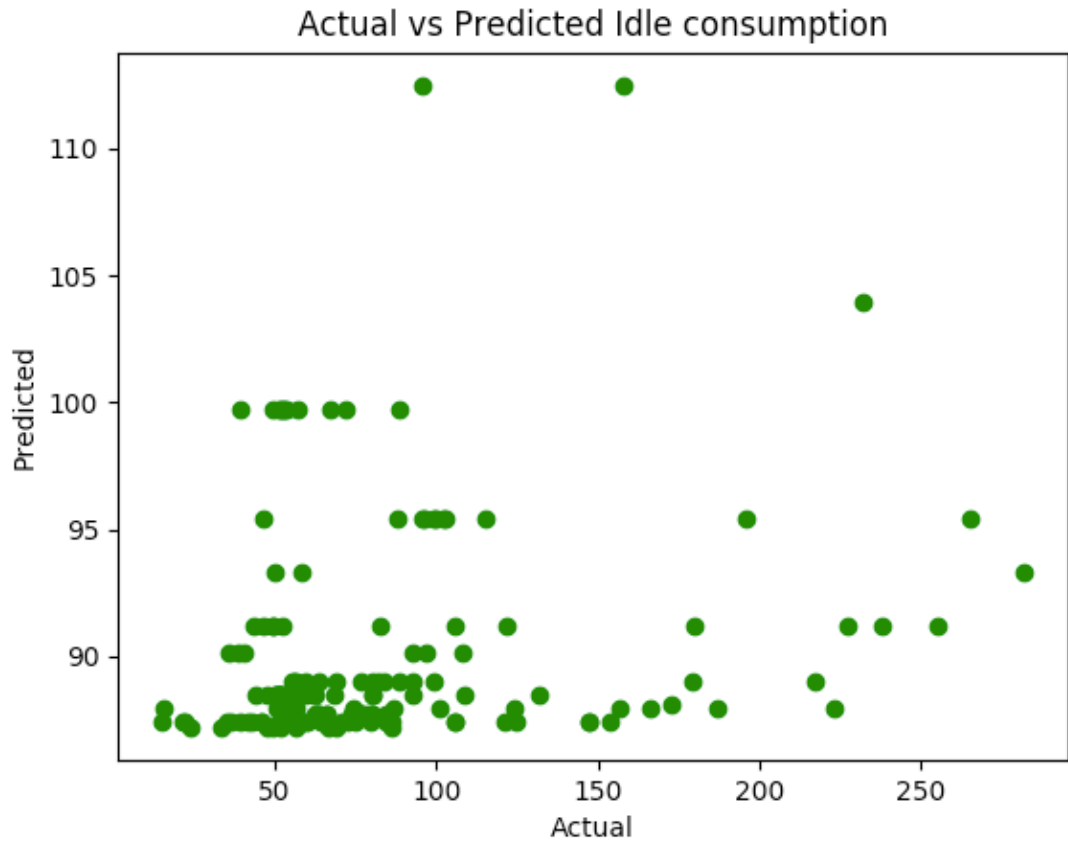


Figure 19: Actual Vs Predicted

Actual Vs Predicted visualised below:

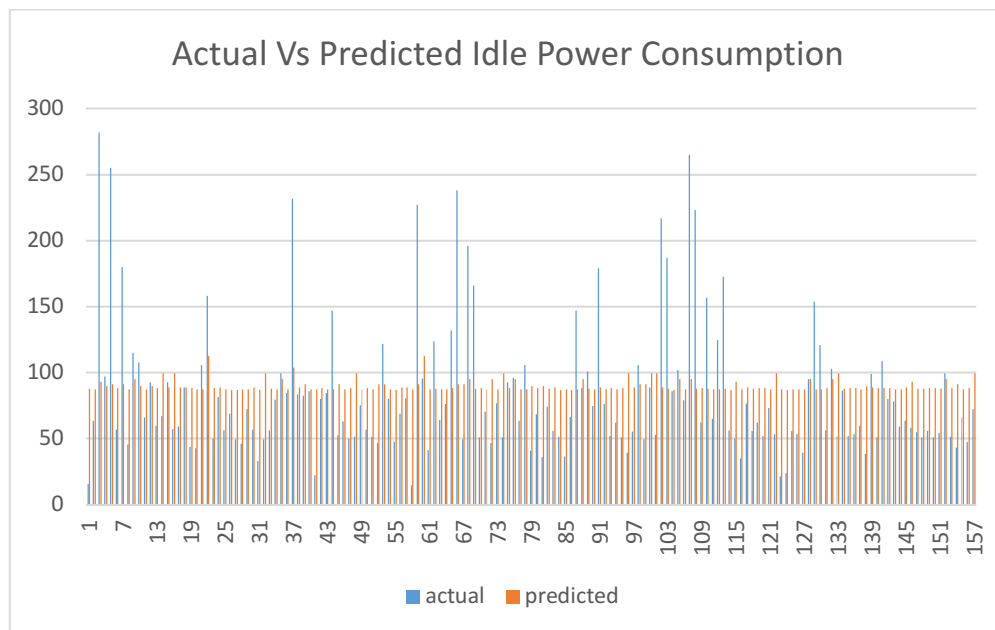


Figure 20: Actual Vs Predicted

The best prediction result for predicting fully utilized power was 29.97 % in SLR and that is when the number of cores is used as the independent variable.

It suggests that SLR is not a good enough technique in predicting power consumption in this context. This means that research should be directed towards a more complex linear regressions method like Multiple Linear Regression (MLR) to see if that will improve prediction accuracy.

4.7 Multiple Linear Regression (MLR) Analysis

As discussed in the above section, SLR using one independent variable is not a sound methodology for predicting energy consumption in this context. So, we started introducing an extra variable.

A linear model was constructed using the three variables as below

X1 = server CPU speed (MHz)

X2 = number of Cores

Y = Predicted utilized energy consumption (in watts)

So, the assumed regression relationship is:

$$Y = b_0 + b_1X_1 + b_2X_2$$

Equation 12 : Multiple Linear Regression

Below, we added number of cores as a second independent variable.

Processor speed and number of cores:

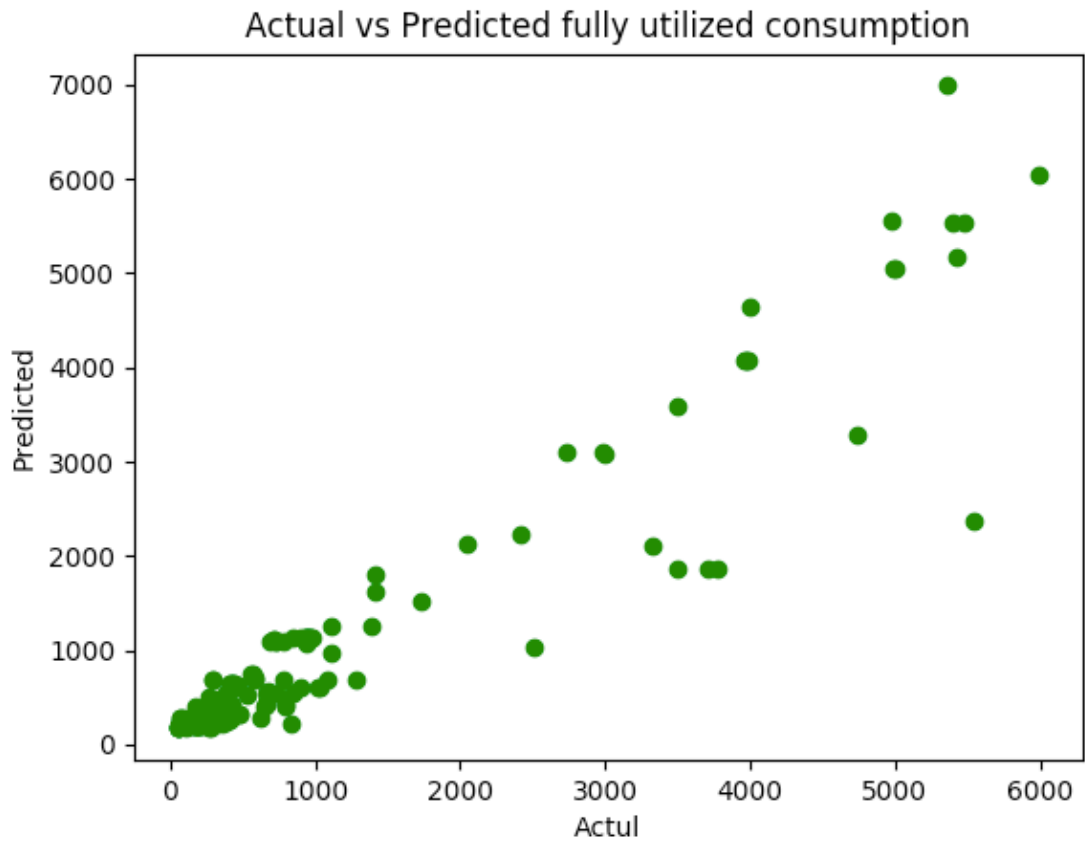


Figure 21:Actual Vs Predicted

Average error %	Intercept	Coefficient of Processor Speed	Coefficient of number of cores
39.98	-20.37	0.07	8.76

We see a significant improvement in the average error percentage.

Below we added one more independent variable
 Processor speed, number of cores, reference age:

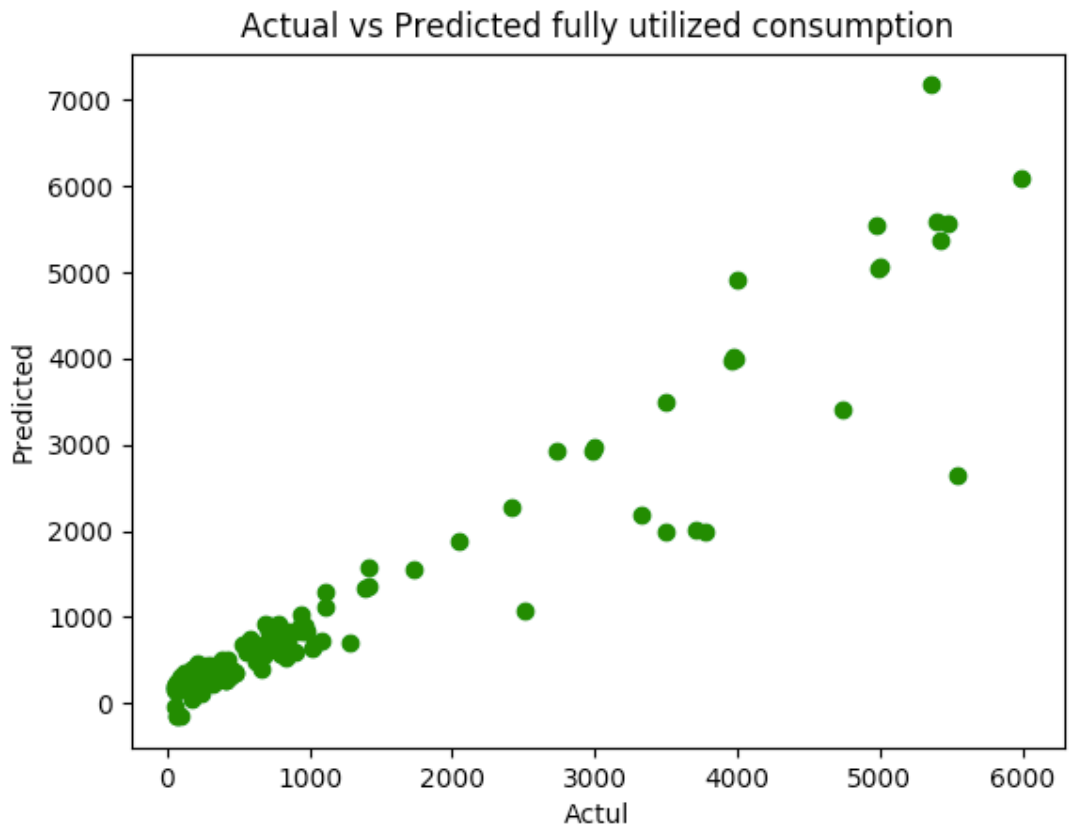


Figure 22:Actual Vs Predicted

Average error %	Intercept	Coefficient of Processor Speed	Coefficient of number of Cores	Coefficient of Reference Age
46.06	618.47	-0.01	9.37	-4.15

Then we added memory to the independent parameter list. The result is below:

Average error %	Intercept	Coefficient of Processor Speed	Coefficient number of Cores	Coefficient Reference Age	Coefficient Memory
46.06	622.37	-0.01	9.30	0.02	-4.17

The memory does not make much difference to the error percentage. This is because the memory does not change much in the data set.

Results are visualised below:

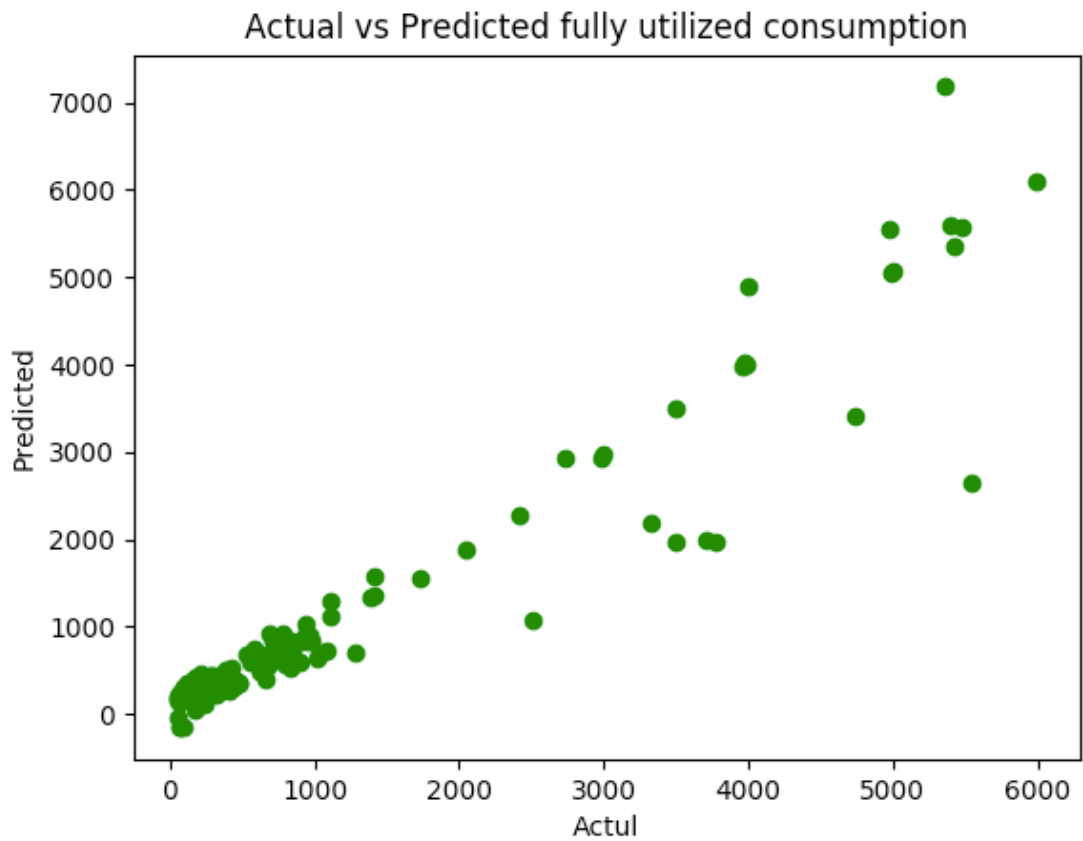


Figure 23: Actual Vs Predicted

4.8 Conclusion on regression model analysis

Based on the analysis on the regression models, we draw below conclusions:

1. for the data set we used, single linear regression has some significant drawbacks as we do not see any strong linear relationship.
2. Multi linear regression produces better prediction accuracy than the single linear regression.
3. IN MLR, the best candidate is the processor speed and the second-best variable is the number of cores in the processor.
4. The error prediction using linear regression is too high to use in a production environment.
5. Memory on its own can be used in the prediction, it does not have much impact on the prediction error when used with other variables.

How we leveraged machine learning to solve the problem of estimating the energy consumption of data centre servers with unknown specifications is discussed in this chapter. We presented the approach of analysing the models and how they were extended to improve the accuracy of the predicted results. How these ML models were implemented as a comprehensive tool and evaluated using a real-world test is discussed in chapter 7.

MALEP v2: DEEP LEARNING PREDICTION

5.1 Introduction

Chapter 4 discussed how ML regression models can be successfully used to predict the server energy consumption. However, there is a significant drawback in the regression models. The average error percentage between the actual and the predicted consumption is too high to be used in a production environment with high level of confidence. Details on the comprehensive regression model evaluation is presented under the evaluation section of this thesis, chapter 8.

In the aim of improving the accuracy, we turned our focus towards deep learning.

5.2 What is Deep Learning?

Deep learning is a subcategory of machine learning and has been a research area for a long time. But it only became popular after the year of 2000, thanks to more powerful computer power and the availability and the capability of handling large memory segments.

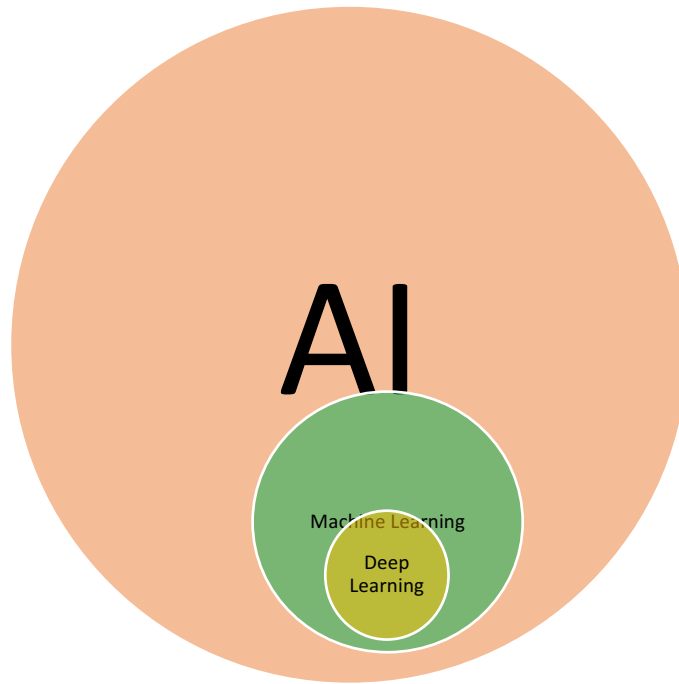


Figure 24 : Deep Learning

DL is sometimes called Hierarchical Learning due to its layered learning structure where the learning output of one layer feeds the next layer as input.

5.2.1 Artificial Neural Networks

ANN simulates human brains, capable of acting like interconnected brain cells which are organized into multiple layers. They are capable of reasoning and making decisions similar to how the human brain would. ANN is able to process the input information and make insights and then delegate the next level of processing to the next layer in line. It is capable of self-learning from input data. Figure 25 represents the organization structure of a neural network:

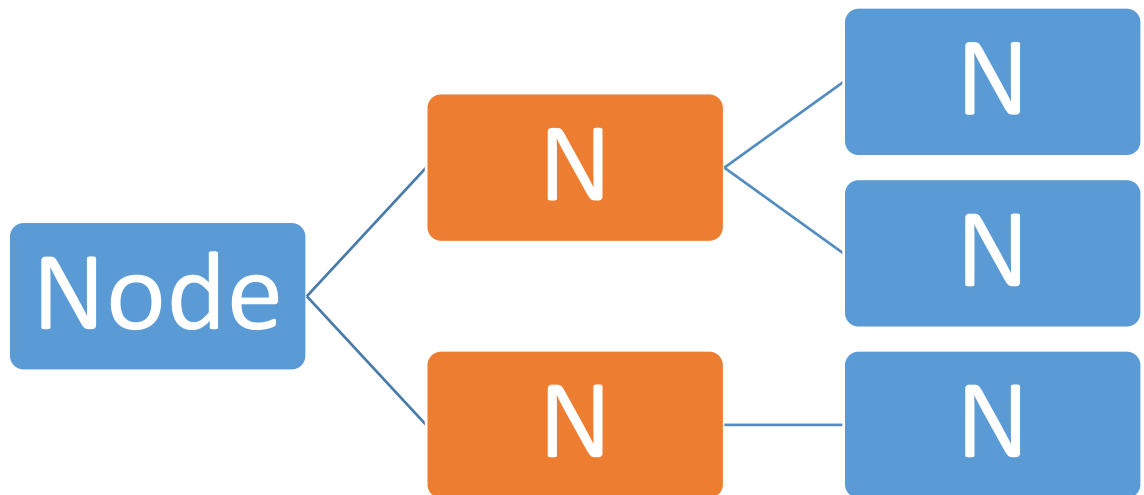


Figure 25 : Neural Network Structure

5.2.2 Deep Neural Networks

DNN is a subset of Artificial Neural Networks. What distinguishes DNN from ANN is the fact that DNN consists of input layers, output layers and deep hidden layers which sit in the middle.

5.3 Why Deep Learning?

Deep Learning is the technique of finding hidden patterns within a large set of data. It opens the door for multi-layered processing models which are capable of learning the patterns in the data in multiple levels of abstraction. The biggest difference between traditional machine learning techniques and deep learning is that in DL, the outcome rules are not defined at the beginning but learned as part of the process.

5.4 Related Deep Learning Techniques

The most popular deep learning technique is the Neural network. The NN is a directed graph with some connected nodes, each taking part in the computations on the flowing data through the model. A node can have zero or many inputs and also zero or many outputs, depending on the model construction configuration.

Figure below depicts typical organization of a neural network:

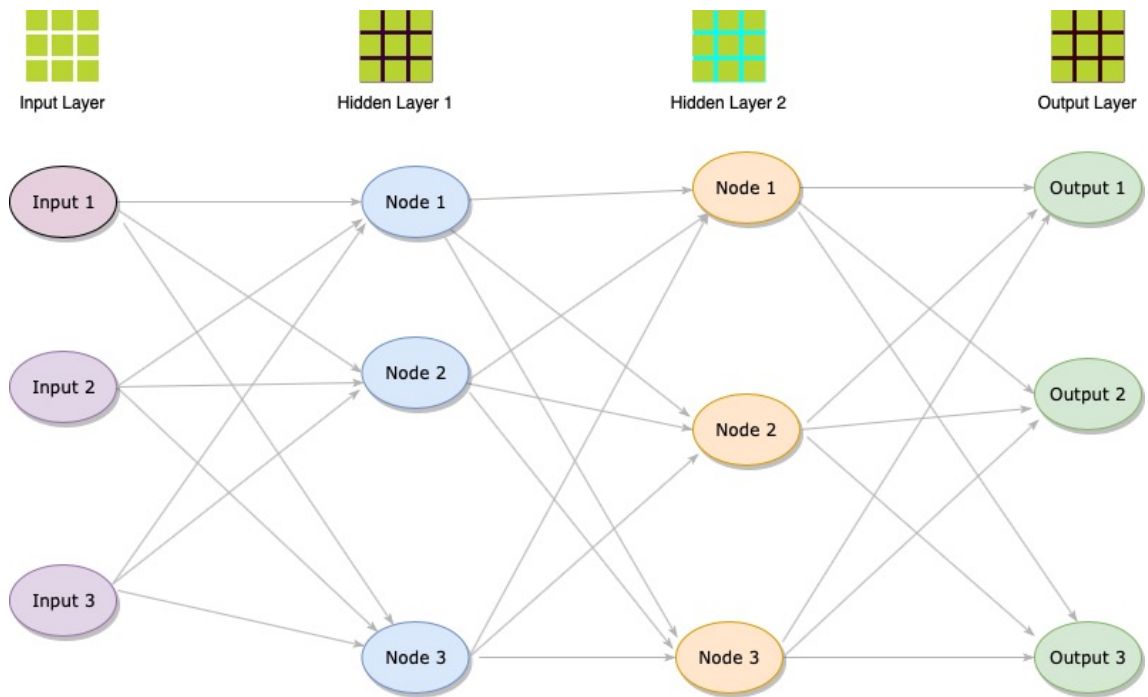


Figure 26: Organization of a Neural Network

5.5 Model Analysis

5.5.1 Feature selection

Deep learning is primarily data driven. So, we studied the list of features available in the chosen dataset to identify the most fitting features for the models. The selected features are listed below.

- Processor Speed
- Number of Cores
- Memory
- Threads per Core
- Reference Age
- Power Supply Rating

The list of features here is same as the list of input parameters used in the linear regression models. Reason for selecting these features was because analysis showed they have some form of relationship with the power consumption.

5.5.2 Models

Deep learning models are built using neural networks. We reviewed the models available in the neural network context to select the most suitable one. Below are the two models available within the framework:

Sequential – It is a stack of linear layers. It allows the construction of model, layer-by-layer, but not sharing layers.

Functional API – More flexible than sequential where a particular layer can be connected to multiple previous or next layers. Fit for creating very complex models like residual network.

After a careful requirement analysis on the model, we decided that we need all our layers to connect to only the previous input and next output. We do not have any requirement for a layer to connect to more than two. So, the sequential model is chosen as the best suitable model for the neural network.

We built the NN sequential model by feeding the input data. Figure 27 depicts structure of that model to predict energy consumption:

Layer (type)	Output Shape	Param #
dense_1 (Dense)	(None, 128)	384
dense_2 (Dense)	(None, 256)	33024
dense_3 (Dense)	(None, 256)	65792
dense_4 (Dense)	(None, 256)	65792
dense_5 (Dense)	(None, 1)	257

Figure 27 : MALEP Neural Network Layers

The model consists of five layers.

Dense is a density-connected neural network layer which abstracts an operation. The number on the output shape of each layer represents the output space dimension e.g. 128 in the dense 1.

We designed the model in a way that the first input layer of the model with an output dimension of 128. That means, the input layer handles the model inputs and emits 128 outputs.

The entire model is visualized in the figure 28 with each dense and associated input and output space dimensions:

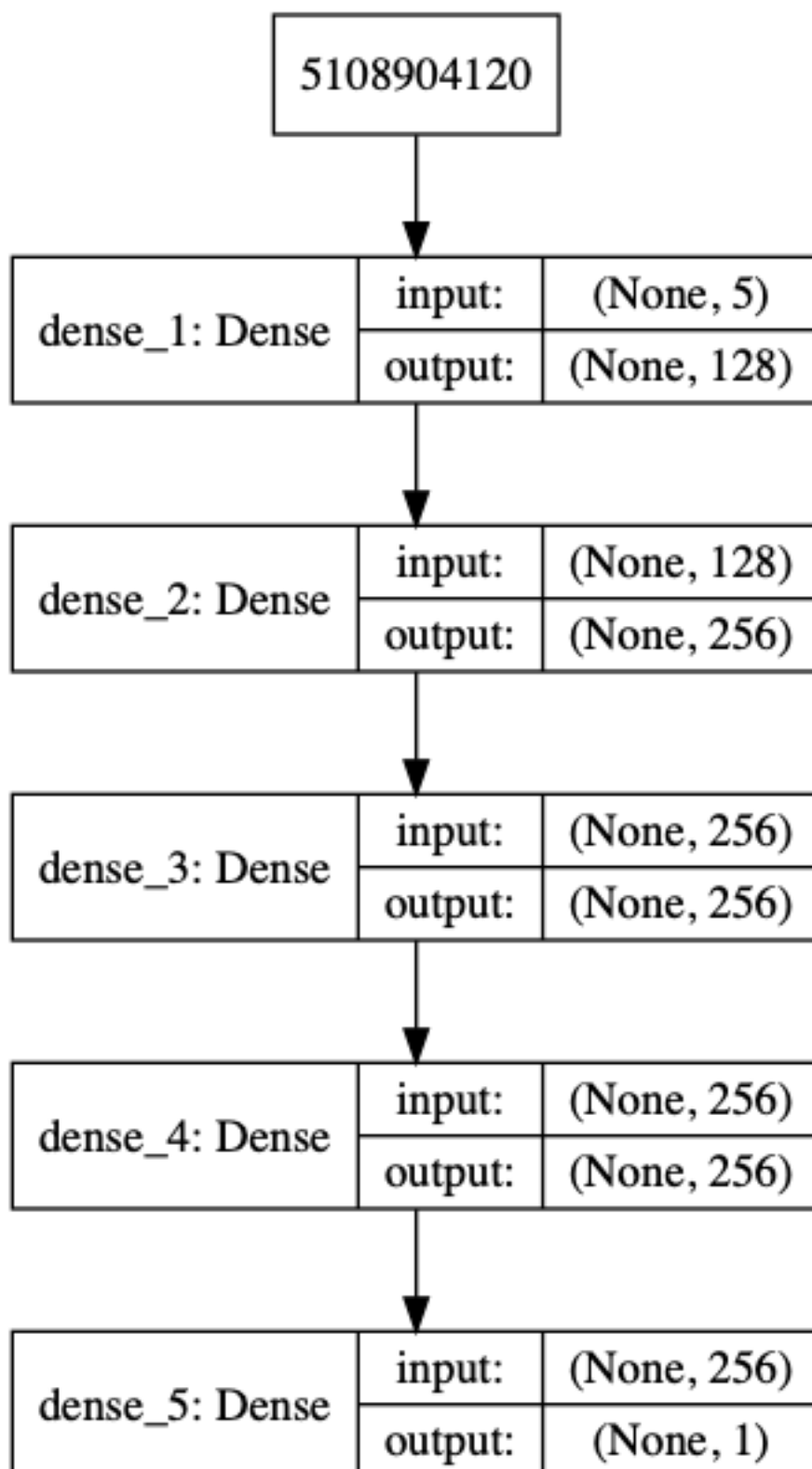


Figure 28 : Visualized Neural Network Model

The technology we utilized for building and testing the model is Python and its libraries. For implementation of core neural network models, we used a library called Keras (keras.io). Keras is based on TensorFlow.

A detailed discussion on the technology is provided in the implementation chapter. Figure 17 is the execution model definition which is visualised using *Tensorboard*(<https://www.tensorflow.org/tensorboard/r1/graphs>), a utility tool comes with Tensorflow for visualizing models. Tensorboard uses the model execution logs to produce the graphs. The visualized model displays how the training data is fed through each individual dense from bottom to top. Figure 29 is the visualized model we constructed for the prediction engine:

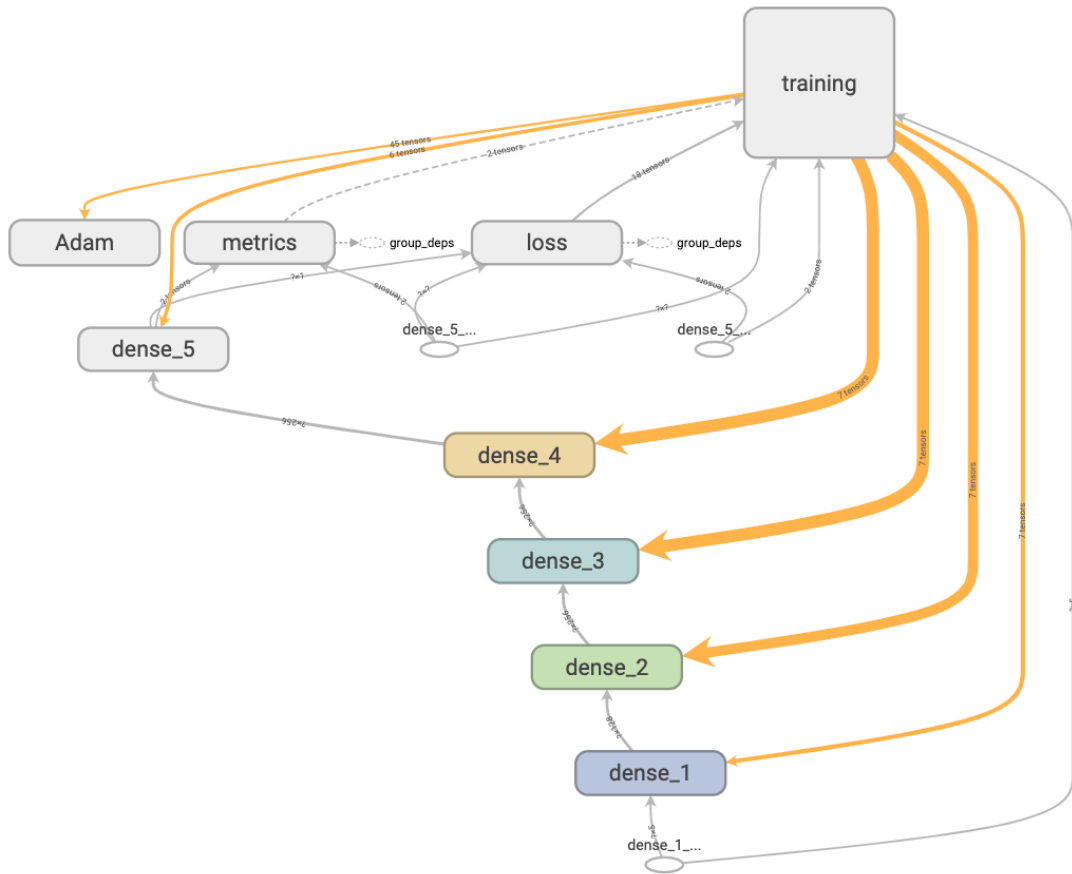


Figure 29 : Dataflow of the Model

5.6 Model Optimization

As the neural network models were now identified, we then moved onto to finding the best optimized configurations to produce the most accurate predictions.

Model: Sequential

Firstly, with the sequential model, there are a few configuration parameters are listed below with the explanations:

1. **Epoch** – the arbitrary cut-off point and one cycle or pass through the entire data set through the model. Epoch is used to separate training into different phases.
2. **Dense** – A layer of connected nodes in a neural network. A dense is constructed by specifying the size of input and output in the form of an array.
3. **Batch size** – a batch approximates the distribution of an input data set. When the batch size is large, the approximation is better, but the drawback is that it takes longer to finish the job and has the risk of running out of memory on the system. So, it is important to find the best batch size which is big enough to predict with acceptable accuracy while maintaining the performance of the process.

As a result of running the models for high number of times, we concluded that the best configuration is below:

ePoch	batch size
500	20

With the optimized configuration, we managed to achieve the best average error rate of **12.38 %**.

The graph below displays the actual against the predicted utilized power consumption for the best optimized configuration:

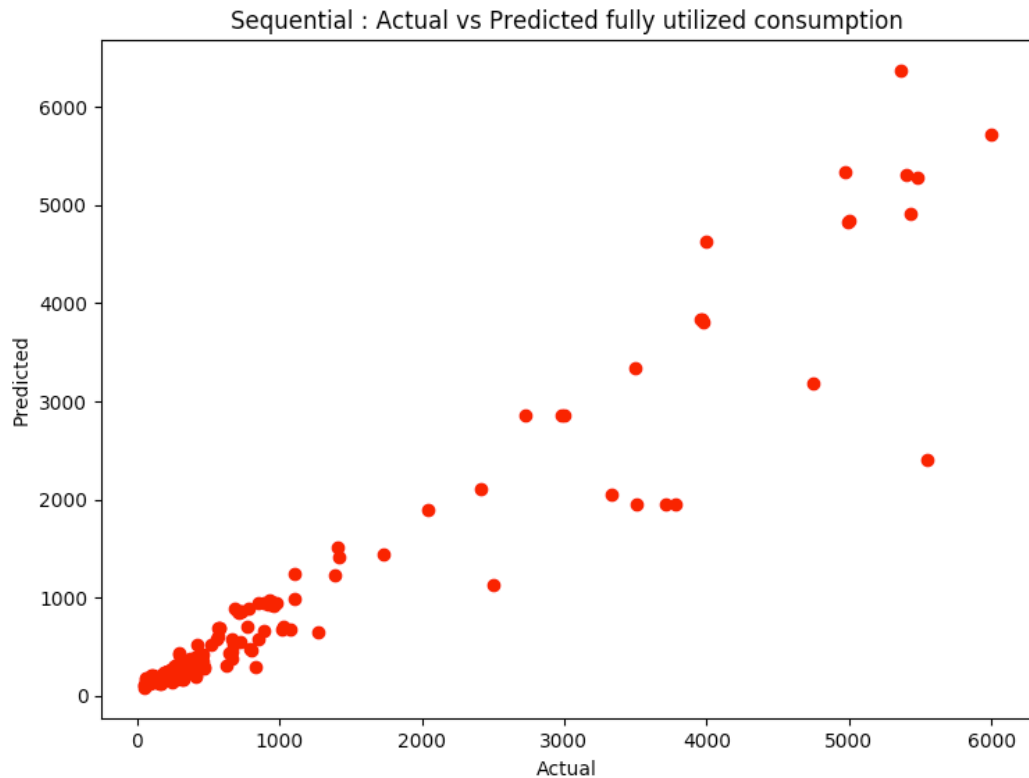


Figure 30 : Sequential Model Actual vs Predicted

5.7 Summary

Since the prediction accuracy of server power consumption using regression models is not good enough for production level, we looked into other machine learning techniques for better results. Deep learning based neural networks were identified as a potential candidate and we carried out model analysis using the same test data as for the regression models in the previous chapter. The visualized models presented better explained layers of the neural network.

Finally, detailed analysis steps to identify the optimized algorithms and configurations are presented. Discussion on how these models were implemented in the MALEP tool are discussed in chapter 7 and evaluation results are in the chapter 8.

Part III

Implementation

“Imagination is more important than knowledge. Knowledge is limited. Imagination encircles the world.”

- *Albert Einstein*

Chapter VI

IMPLEMENTATIONS OF CALCULATION MODELS

6.1 Introduction

In chapter 3, we presented the models for calculating the total energy for each equipment types in data centres using the idle and utilized consumption of individual equipment. This chapter introduces how those models were implemented as an independent new module and how it was integrated into the main Eureca project. The application can be accessed in - <https://tool.dceureca.eu/#!/Calculator>

6.2 Application Architecture

The application was designed using Model View Controller (MVC) design pattern. Figure 31 illustrates the basic principle of MVC:

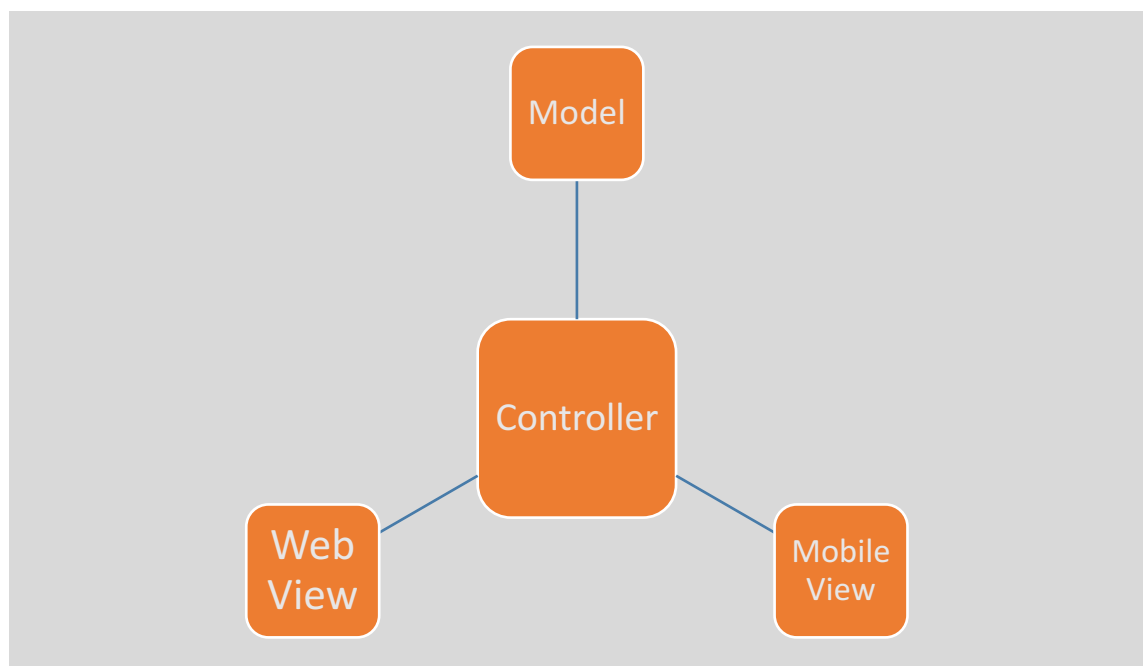


Figure 31: Application Architecture

The Model: Represents the data and the interaction with the database which includes both data updates and reads.

The controller: Connects the model and the view. It also contains the logic for request and response handling, transfer data between the view and the model.

The View: Is the user-interactive layer, which is responsible for the presentation. It is the user interface and in addition to the presentation, it helps to capture and validate user inputs.

Open-source framework Vaadin (<https://vaadin.com/>) was used in the MVC pattern-based implementation in the tool. The componentised nature of Vaadin makes the application configuration easier. It also helps to reduce boilerplate code which is required to bring up the necessary scaffolding of the application.

6.2.1 Other Design Patterns

Apart from MVC, the other designed patterns used in the application design are listed below:

- **Data Transfer Object (DTO)**

Data is wrapped in Objects are transferred between application tiers. Plain java objects are constructed and populated either at the data access layer or the presentation layer and propagated up or down the stack.

- **Data Access Object (DAO)**

DAO pattern was used for handling and encapsulating database interaction of the application. While data retrieval and update are its main responsibility, other non-functional aspects like improving the performance through connection pooling are also fulfilled in this layer.

- **Multi-Threading Paradigm**

The application was designed and developed as a highly scalable and performance driven way which benefits the users by providing a faster access experience. The multiple cores in the processors were utilized by running

multiple user sessions concurrently. How multiple user sessions are live in-parallel severing the same function is represented in the below figure 32:



Figure 32 :Multi-threaded sessions

6.3 Technology

This section details the technologies used in the implementation of the tool.

6.3.1 Java & EcoSystem

Java was the primary programming language chosen in the tool implementation. The reason for that decision was to be in line with the existing Eureka platform and the consistency.

Java has a vibrant ecosystem with numerous open-source frameworks and libraries available to complement the power of Java in the development and production environments. Table 8 lists the related technologies and framework we utilized in the calculation model tool with the main the purpose of being used:

Framework	Purpose	Reference
Vaadin	MVC framework	vaadin.com
JSTL	Tag library for UI	docs.oracle.com/javase/5/jstl/1.1/docs/tlddocs/
Maven	Build tool	maven.apache.org
JDBC	Database connectivity	docs.oracle.com/javase/8/docs/technotes/guides/jdbc/
MySQL	Database	www.mysql.com
JUnit	Unit testing	junit.org/junit5/
Mockito	Mocking backend objects	site.mockito.org
GIT	Version controlling system	git-scm.com

Table 8 :Java Frameworks used in the Application

6.3.2 VAADIN

Vaadin is an implementation framework of MVC pattern and it is written in Java. That makes it seamlessly integrate with the application. The framework provides the template for application components to abstracts the behaviours easily and quickly. That enabled us to spend most of our development time on solving business problems as the basic application scaffolding templates had already been provided by the framework.

6.3.3 Maven

Since the application integrates with a number of libraries and frameworks like Vaadin and JDBC, managing the dependencies became a complicated and tedious piece of work. To address this problem, we decided to use a dependency management tool. The candidates we considered and evaluated were Ant, Maven and Gradle. However, Maven was the winner due to easier integration and the ability to maintain the dependencies for all environments like development, testing and production.

Project Object Model or POM file was created by listing all dependent libraries with their inter dependencies and versions. The format of POM file is xml. When the project was built for the first time, Maven reads the POM file and downloads all required

dependencies from the sources and then stores them in a local repository. For subsequent builds, it uses the libraries from the local repository reducing the building time. It has helped in organizing the project dependencies along with their respective versions as well as improving the efficiency of the build process. Figure below is an extract from the POM file we have used.

```
<dependency>
  <groupId>com.lambdaworks</groupId>
  <artifactId>scrypt</artifactId>
  <version>1.4.0</version>
</dependency>

<dependency>
  <groupId>org.vaadin.addon</groupId>
  <artifactId>confirmdialog</artifactId>
  <version>2.1.3</version>
</dependency>

<dependency>
  <groupId>org.vaadin.addons</groupId>
  <artifactId>vaadin-combobox-multiselect</artifactId>
  <version>1.1.14</version>
</dependency>

<dependency>
  <groupId>org.vaadin.addons</groupId>
  <artifactId>justgage</artifactId>
  <version>1.0.1</version>
</dependency>

<dependency>
  <groupId>org.apache.commons</groupId>
  <artifactId>commons-email</artifactId>
  <version>1.4</version>
</dependency>

<dependency>
  <groupId>org.vaadin.addons</groupId>
  <artifactId>screenshot</artifactId>
  <version>0.4.1.2</version>
</dependency>
```

6.3.4 Version Controlling & Code Repository

Version controlling is the process of keeping a different version of the code at the file level, which enables us to build and deploy a certain version of the application. GIT was the version controlling tool we used.

The code repository is the centralized place where the code is securely hosted. We used BitBucket (<https://bitbucket.org>) as code a hosting and sharing repository.

6.3.5 Environment Management

The application is available in multiple environments, as listed in table 9. The code is written, and unit tested in the development environment. It is then deployed to a test environment for integration and smoke testing. Once everything is good in the testing environment, the application is finally released to the production environment for users to access.

Environment	Purpose
Development	Application Development and Unit testing
Testing	Integration testing and pre-production sanity checks
Production	Deploy the application for users

Table 9 :Application Environments

Figure 33 demonstrates how the application code is promoted through each environment with the main responsibility and purpose is mentioned:

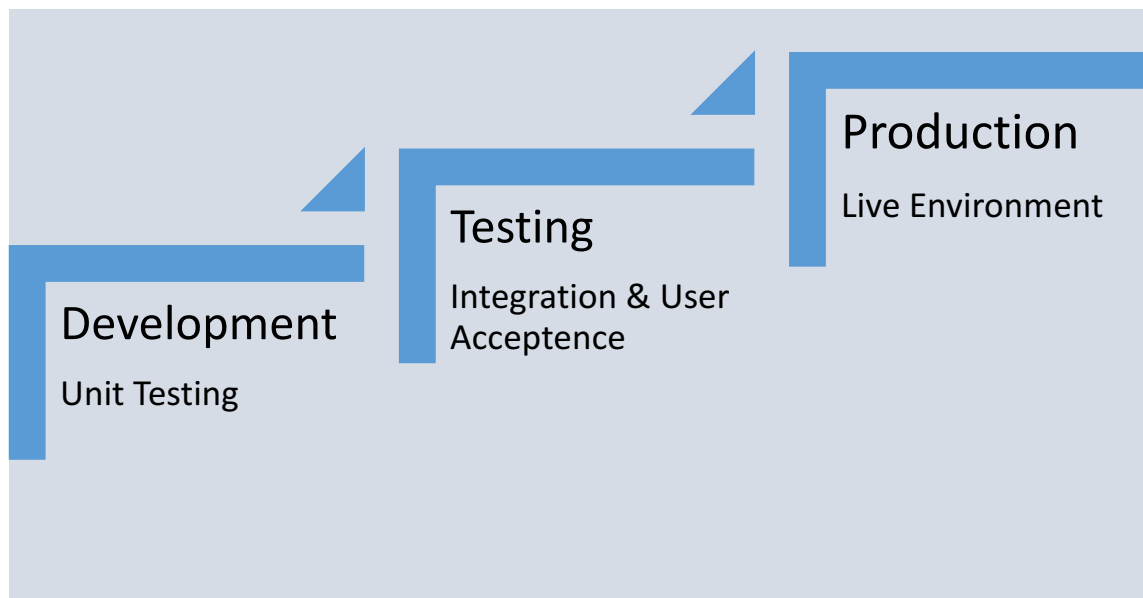


Figure 33:Code Promotion Steps

6.3.6 Unit Testing and Test-Driven Development

Unit testing is an integral part of software development as it helps to detect bugs in the early stage and increase the developer’s confidence on later refactoring tasks. We used the Open-source framework *Junit* for writing test cases to cover individual functionality.

Test Driven Development (TDD) was the development process we followed. It helped us to understand the requirement better to write our test cases. Having a good test coverage gave us the confidence to carry further refactoring work. As an example, if any test case in the test-pack fails, that would indicate that the new code we committed to the repository was likely to have introduced a defect.

Figure 34 below explains the flow of TDD:

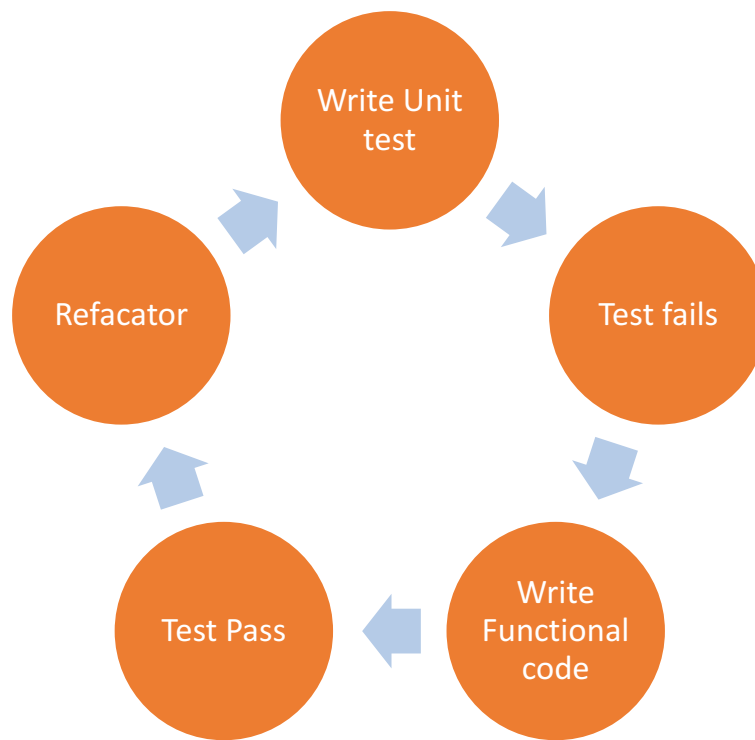


Figure 34 :TDD Cycle

Before writing any functional code, we wrote the unit test case for the functionality in question. This particular test case was expected to fail immediately because no associated functional code is written yet. Then we wrote bare minimum code, just to pass the test and then refactored the functionality by bringing in more functional code.

The same cycle applied multiple times to increase the functionality and test coverage. Given below is a code example of a test case from the tool. It calculates the server energy calculation model for given utilized power consumption, idle power consumption and utilization percentage.

If this particular test case fails, when all the test cases in the package have been executed, we then know immediately that a defect is introduced. We know we need to revisit the most recent changes as they may possibly have a knock-on effect on the server energy calculation function.

Shown below is some code from an application unit test case:

```

1. @Test
2.     public void calculateServerEnergy() {
3.         double expected = 194760;
4.
5.         double actual = calculatorUtility.calculateTotalServerEnergy(12,370, 1
6.             10,62);
7.
8.         assertNotNull(actual);
9.         assertEquals("Error in calculated server energy", expected, actual, 0)
10.        ;
11.    }

```

It tests the application functionality by executing it with some input. The code then compares the expected value with the actual output and fails the test with an error message, if the condition under test is not met.

6.3.7 Deployment

The tool is deployed on Linux in a dedicated space in the Eureka multi-tenant server. The application code is compiled and bundled into an executable web archive file type called – WAR file. Since it is a web application, the war file is submitted into the webserver so it can be deployed on it. The webserver we used for deployment is Apache Tomcat.

6.4 Workflow of the Tool

This section provides an overview of the system workflow, how a particular data centre operator could use it to estimate the total and individual equipment type energy consumption.

6.4.1 Energy Calculator

As shown in figure 35 which is an input screen, a user can create the data centre equipment portfolio by choosing the brands and product types, the number of each type etc. This is where the system captures all the required input parameters for the model execution.

Energy Calculation

Country *
Ireland

Server Equipment *	No of servers *	Server utilized(100%) Power(Watts) *	Server Idle Power Rate(Watts) *	Server utilization(0-1) *
Cisco UCSC210 A	10	301	125	0.2

UPS Equipment *	No of UPS *	UPS Power Rating(Watts) *	UPS Utilization(0-1) *	UPS Efficiency(0-1) *
CAPC Smart-UPS	2	2700	0.5	0.93

Storage Equipment(Disk) *	No of Storage *	No of Disk drivers *	Disk Power Rate(watts) *	Disk utilization(%) *
CEMC VNXe3300	5	12	450	0.8

Cooling Equipment *	No of Cooling Units *	Cooling Power Rate(watts) *	Cooling utilization(%) *
TOSHIBA Air Cor	4	5500	1

Network Equipment *	No of Network Equipment *	Network Power Rating(Watts) *	Network Utilization(0-1) *
Cisco C2960G IS	6	120	1

Other Cost Involved

Annual cost of Licensing *	Annual cost of Installation *	Annual cost of Staffing (Facility) *	Annual cost of Staffing (Facility) *	Annual cost of Rental *
4000	6000	12000	12000	15000

Calculate

Total Annual cost of Energy (Electricity)

Figure 35 :Input Screen

6.4.2 Output Screen

After setting up the equipment portfolio, a user can kick off the calculation process.

The system executes the models and present the results in the output screen shown as in figure 36:

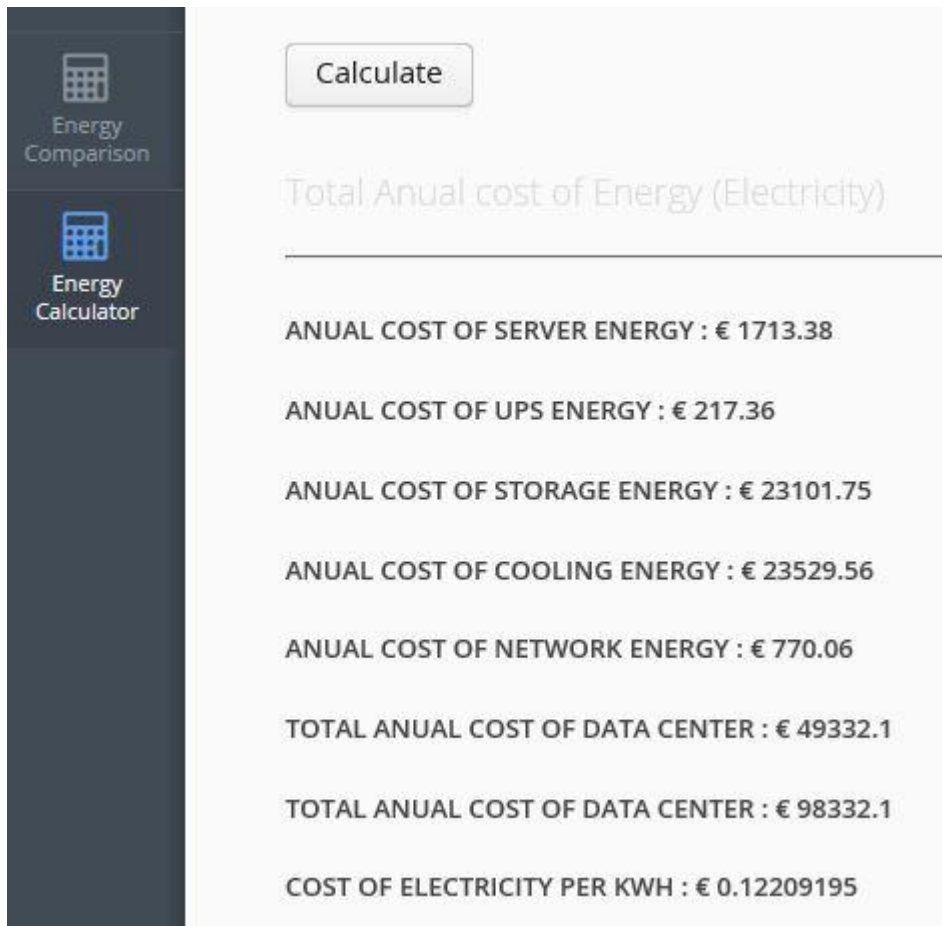


Figure 36 :Output Screen

6.4.3 Energy Snapshots

An energy snapshot is the state of energy portfolio at a given time. New snapshots are created by changing different parameters of the portfolio. One such example would be by changing the equipment which has different energy efficiencies. The energy snapshots are generated by capturing the details on each equipment types and saving them in the system. This functionality enables the data centre operators to compare the outcome of improvements made to their energy profile.

Figure 37 shows the generic snapshot details capturing screen where the beginning of the workflow process to drive the user through the energy snapshot construction:

Snapshot Details

Snapshot Servers Network Storage Cooling

DC Site * 19

Snapshot Name * test 123

Type * Baseline

Are Server Details Available * Y

Are Cooling Details Available * Y

Are UPS Details Available * Y

Are Network Details Available * Y

Figure 37 Energy Snapshot

The details captured through the page is then stored in the system to re-construct and render to the user on demand. Figure 38 is the screen of the database table structure to save generic snapshot details:

#	Name	Type
1	id 	int(11)
2	name	varchar(50)
3	organisation	int(11)
4	user	int(11)
5	dcSite	int(11)
6	snapshotType	varchar(255)
7	date	date
8	serverDetails	varchar(1)
9	coolingDetails	varchar(1)
10	upsDetails	varchar(1)
11	networkDetails	varchar(1)
12	pueValue	varchar(40)
13	totalElectricity	varchar(40)
14	itElectricity	varchar(40)
15	upsOutput	varchar(40)
16	itSize	varchar(40)
17	inletTemperature	int(40)

Figure 38 :Snapshot Table Structure

Servers

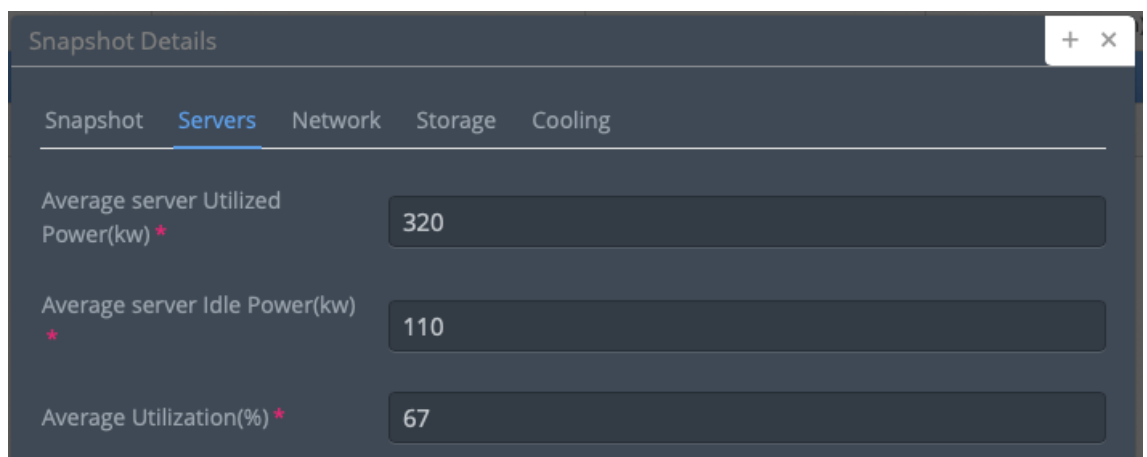


Figure 39 :Server Energy

Database Table Structure


#	Name	Type
1	server_id 	int(40)
2	make	varchar(80)
3	model	varchar(80)
4	power_idle	int(11)
5	power_util	int(11)
6	util_percentage	decimal(40,0)

Figure 40 : Server Energy Table Structure

Network

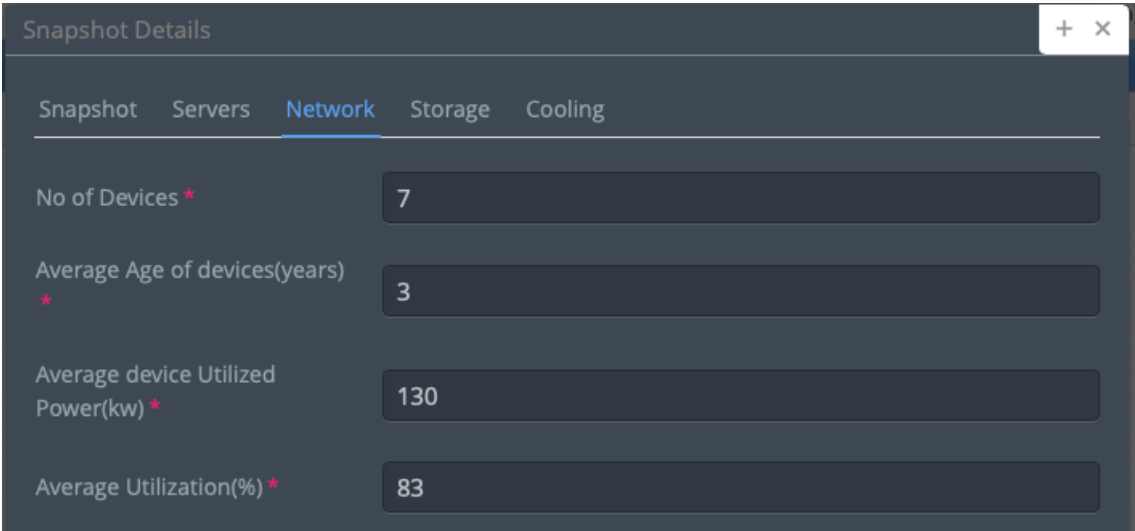


Figure 41 :Network Energy

Database Table Structure



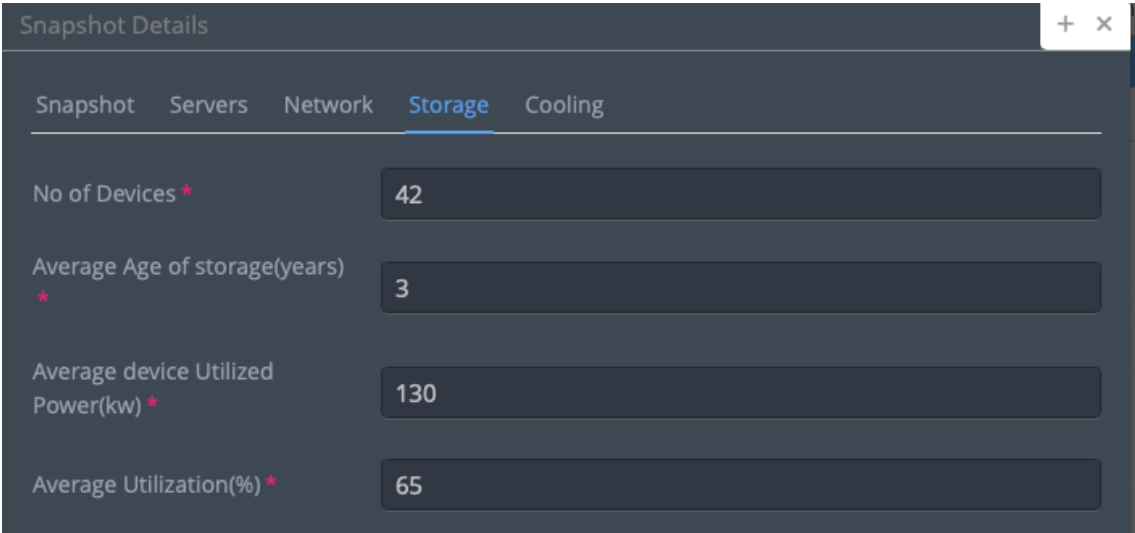
#	Name	Type
1	network_id  	int(40)
2	make	varchar(80)
3	model	varchar(80)
4	power_rate_util	int(11)
5	power_rate_idle	int(40)

Figure 42 :Network energy Table Structure

Storage



The screenshot shows a 'Snapshot Details' window with a dark theme. It has a title bar with a '+' and 'x' icon. Below the title bar are tabs for 'Snapshot', 'Servers', 'Network', 'Storage', and 'Cooling'. The 'Storage' tab is selected. The main area displays four metrics, each with a label and a value in a dark input field:

- No of Devices * : 42
- Average Age of storage(years) * : 3
- Average device Utilized Power(kw) * : 130
- Average Utilization(%) * : 65

Figure 43 : Storage Energy

Database table structure

#	Name	Type
1	storage_id	int(40)
2	name	varchar(80)
3	make	varchar(80)
4	model	varchar(80)
5	power_util	int(11)

Figure 44: Network Energy Table Structure

Cooling

The screenshot shows a 'Snapshot Details' window with a dark theme. It has a tabbed interface with 'Snapshot', 'Servers', 'Network', 'Storage', and 'Cooling'. The 'Cooling' tab is active. It displays two input fields: 'Average Cooling Utilized Power(kw) *' with the value '340' and 'Average Utilization(%) *' with the value '55'. At the bottom, there are three buttons: 'Cancel' (pink), 'Delete' (pink), and 'Save' (blue).

Figure 45 :Cooling Energy

Database Table Structure



#	Name	Type
1	cooling_id  	int(40)
2	make	varchar(80)
3	model	varchar(80)
4	power_util	int(11)

Figure 46 : Cooling energy table structure

Calculation results

The models were executed by feeding the input data, and then the tool calculates the total energy, IT energy and PUE value. The output details are presented in the calculation results dashboard, which is shown in the below figure:

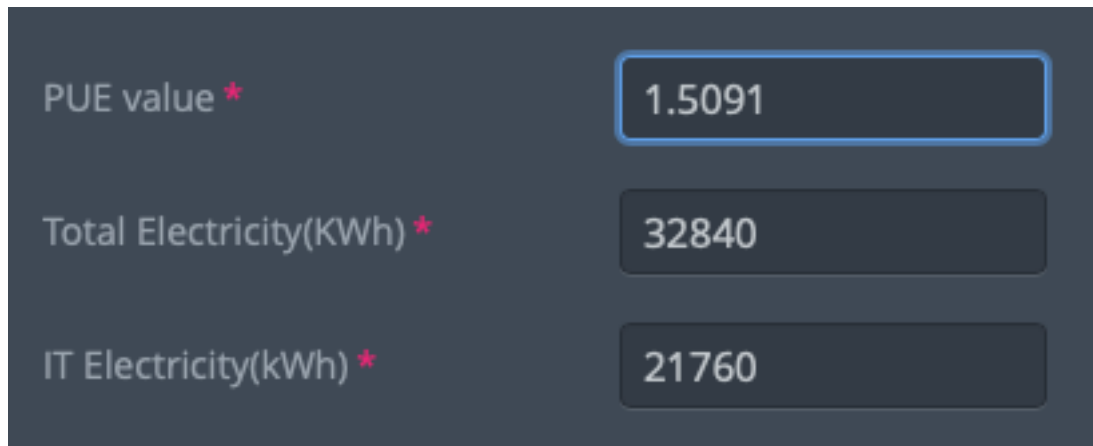


Figure 47:Energy Calculation Results

6.5 Application Security

We designed the tool in a way that it could utilize the existing security service of the main Eureka tool. A handler was introduced to validate each user session through the authentication and authorisation services. It is a common aspect which is applied before any user service request is served. Un-authenticated user sessions are re-directed to login page of the main tool.

6.6 Code Samples

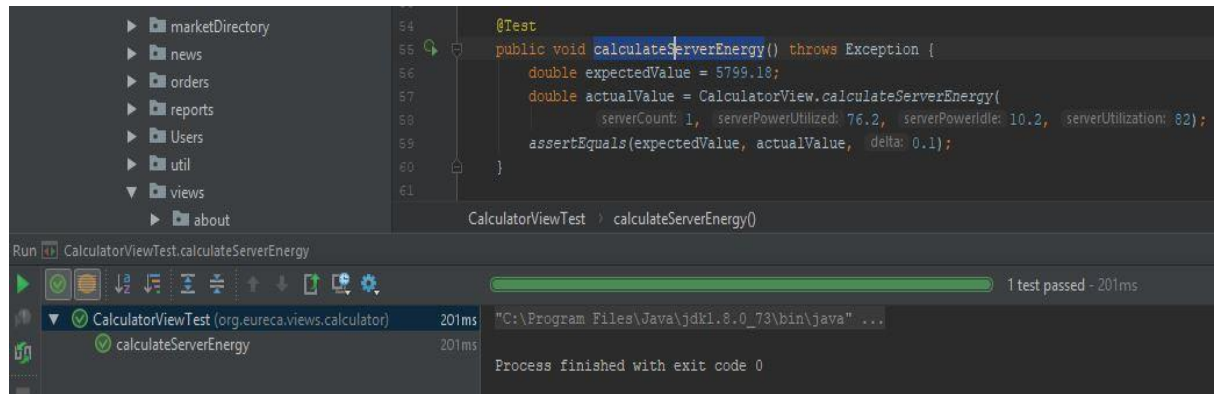
The primary programming language of the tool is Java. Selected Java code samples from the main components are demonstrated here.

6.7 Model Evaluation

The first round of Model evaluation was done using some sample data. That includes both unit testing as well as integration testing. Unit testing details and evidence for each category are discussed below.

JUnit framework (<https://junit.org>) was used for implementing unit test cases.

6.7.1 Server Power Consumption Calculation



```
54
55
56
57
58
59
60
61
@Test
public void calculateServerEnergy() throws Exception {
    double expectedValue = 5799.18;
    double actualValue = CalculatorView.calculateServerEnergy(
        serverCount: 1, serverPowerUtilized: 76.2, serverPowerIdle: 10.2, serverUtilization: 82);
    assertEquals(expectedValue, actualValue, delta: 0.1);
}
```

Run CalculatorViewTest.calculateServerEnergy

1 test passed - 201ms

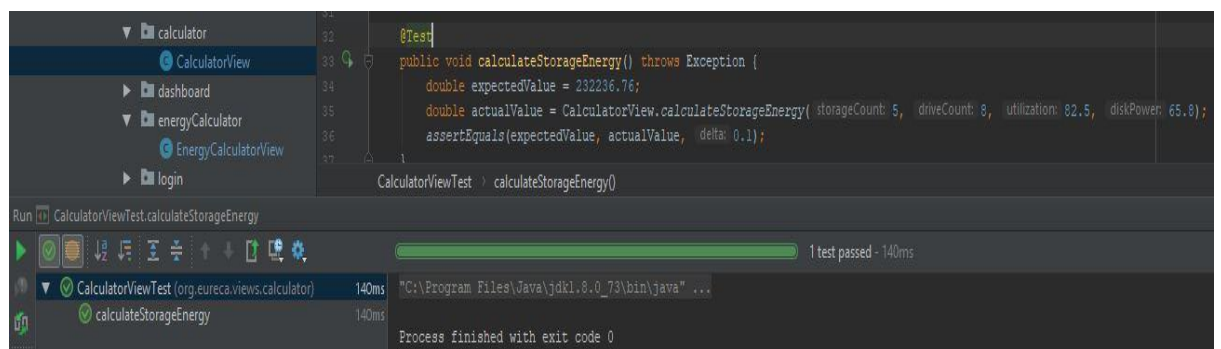
CalculatorViewTest (org.eureca.views.calculator) 201ms "C:\Program Files\Java\jdk1.8.0_73\bin\java" ...

calculateServerEnergy 201ms

Process finished with exit code 0

Figure 48: Evaluation of Server Power consumption

6.7.2 Storage Power Consumption Calculation



```
32
33
34
35
36
37
@Test
public void calculateStorageEnergy() throws Exception {
    double expectedValue = 232236.76;
    double actualValue = CalculatorView.calculateStorageEnergy( storageCount: 5, driveCount: 8, utilization: 82.5, diskPower: 65.8);
    assertEquals(expectedValue, actualValue, delta: 0.1);
}
```

Run CalculatorViewTest.calculateStorageEnergy

1 test passed - 140ms

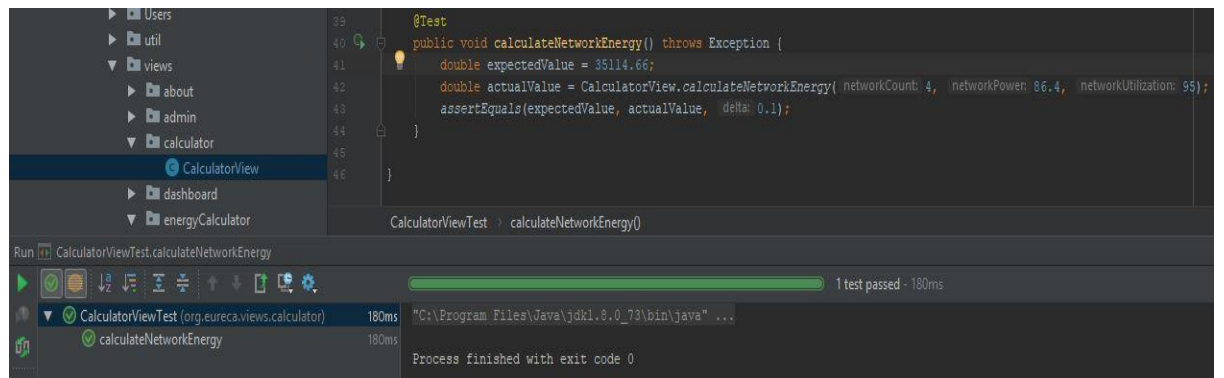
CalculatorViewTest (org.eureca.views.calculator) 140ms "C:\Program Files\Java\jdk1.8.0_73\bin\java" ...

calculateStorageEnergy 140ms

Process finished with exit code 0

Figure 49: Evaluation of Storage s Power consumption

6.7.3 Network Power Consumption Calculation



```
@Test
public void calculateNetworkEnergy() throws Exception {
    double expectedValue = 35114.66;
    double actualValue = CalculatorView.calculateNetworkEnergy( networkCount: 4, networkPower: 86.4, networkUtilization: 95);
    assertEquals(expectedValue, actualValue, delta: 0.1);
}
```

Run CalculatorViewTest.calculateNetworkEnergy

1 test passed - 180ms

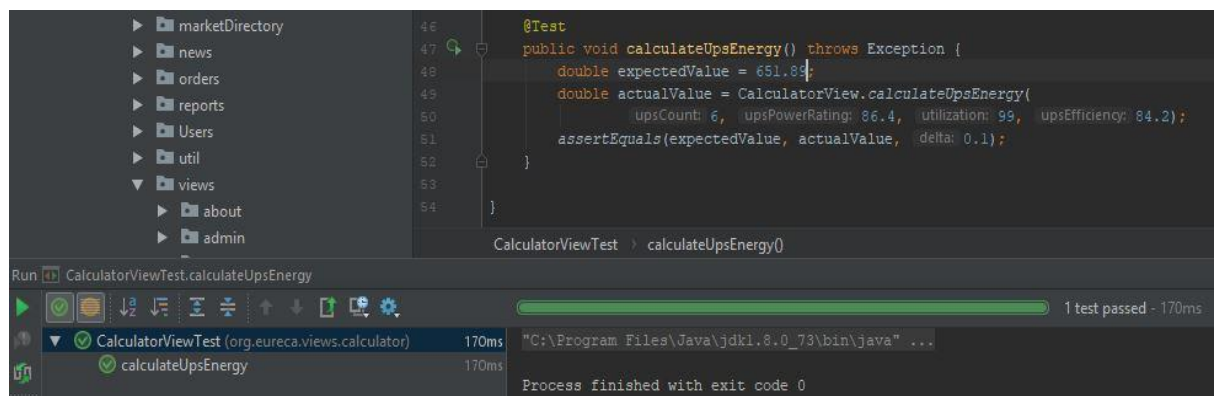
CalculatorViewTest (org.eureca.views.calculator) 180ms "C:\Program Files\Java\jdk1.8.0_73\bin\java" ...

calculateNetworkEnergy 180ms

Process finished with exit code 0

Figure 50:Evaluation of Network Power consumption

6.7.4 UPS Power Consumption Calculation



```
@Test
public void calculateUpsEnergy() throws Exception {
    double expectedValue = 651.84;
    double actualValue = CalculatorView.calculateUpsEnergy(
        upsCount: 6, upsPowerRating: 86.4, utilization: 99, upsEfficiency: 84.2);
    assertEquals(expectedValue, actualValue, delta: 0.1);
}
```

Run CalculatorViewTest.calculateUpsEnergy

1 test passed - 170ms

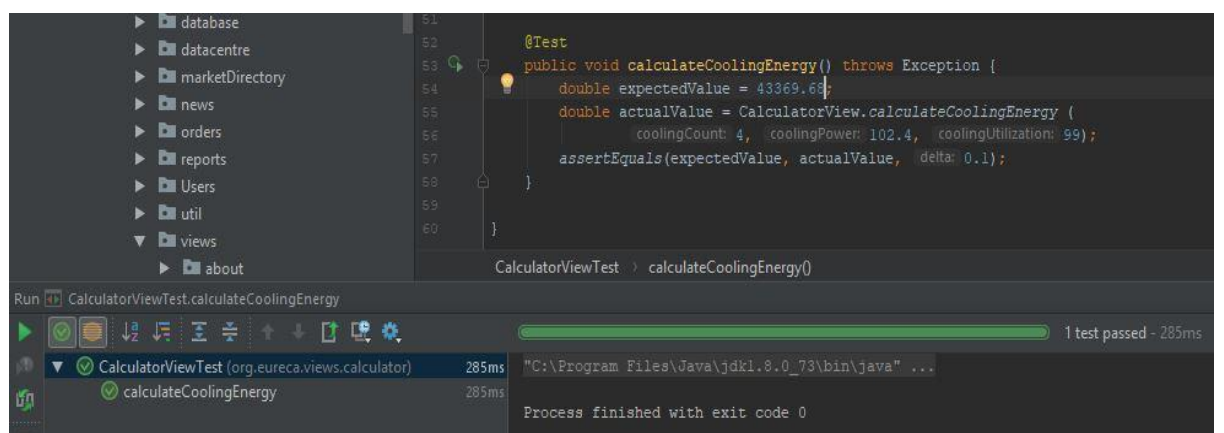
CalculatorViewTest (org.eureca.views.calculator) 170ms "C:\Program Files\Java\jdk1.8.0_73\bin\java" ...

calculateUpsEnergy 170ms

Process finished with exit code 0

Figure 51:Evaluation of UPS Power Consumption

6.7.5 Cooling Power Consumption Calculation



```
@Test
public void calculateCoolingEnergy() throws Exception {
    double expectedValue = 43369.68;
    double actualValue = CalculatorView.calculateCoolingEnergy (
        coolingCount: 4, coolingPower: 102.4, coolingUtilization: 99);
    assertEquals(expectedValue, actualValue, delta: 0.1);
}
```

Run CalculatorViewTest.calculateCoolingEnergy

1 test passed - 285ms

CalculatorViewTest (org.eureca.views.calculator) 285ms "C:\Program Files\Java\jdk1.8.0_73\bin\java" ...

calculateCoolingEnergy 285ms

Process finished with exit code 0

Figure 52:Evaluation of Cooling Power Consumption

6.8 Conclusion

This chapter was relating to the web application we built for data centre operators. They can create and maintain their energy snapshots which reflect the current equipment portfolio and the energy consumption. The tool implements the models discussed in chapter 3 and present the users the opportunity to calculate the current consumption. They can also use the system to preview the impact of any improvements or changes by comparing snapshots.

We also discussed how we evaluated each model with sample data through a testing strategy with unit and integration test cases.

Chapter VII

IMPLEMENTATIONS OF MALEP

7.1 Introduction

This chapter presents the implementation details of the tool for both Multi Linear Regression and Deep Learning-based predictions of server energy consumption.

The tool was built as a modularised enterprise application, applying the latest architectural patterns like Microservices, Restful web services and also utilising technologies like Python, TensorFlow, Keras, SkLearn, Flask, Linux. The tool executes ML algorithms on training data which were obtained from a reliable source in the public domain and predicts the energy consumption for specifications.

It has some multi-user interaction interfaces like web and desktop UI. The application is highly configurable which provides the users with a great level of flexibility. The application provides integration web service endpoints for external applications to consume the service. The chosen message exchanging format is JASON due to its simple but flexible structure as well as being the single most used format in the industry.

The design and architecture of the application are discussed in the next subsection, outlining how various components are organised to serve multi-channel offerings.

7.2 Design & Architecture

7.2.1 Overall View

The Architecture of the application is depicted in figure 53:

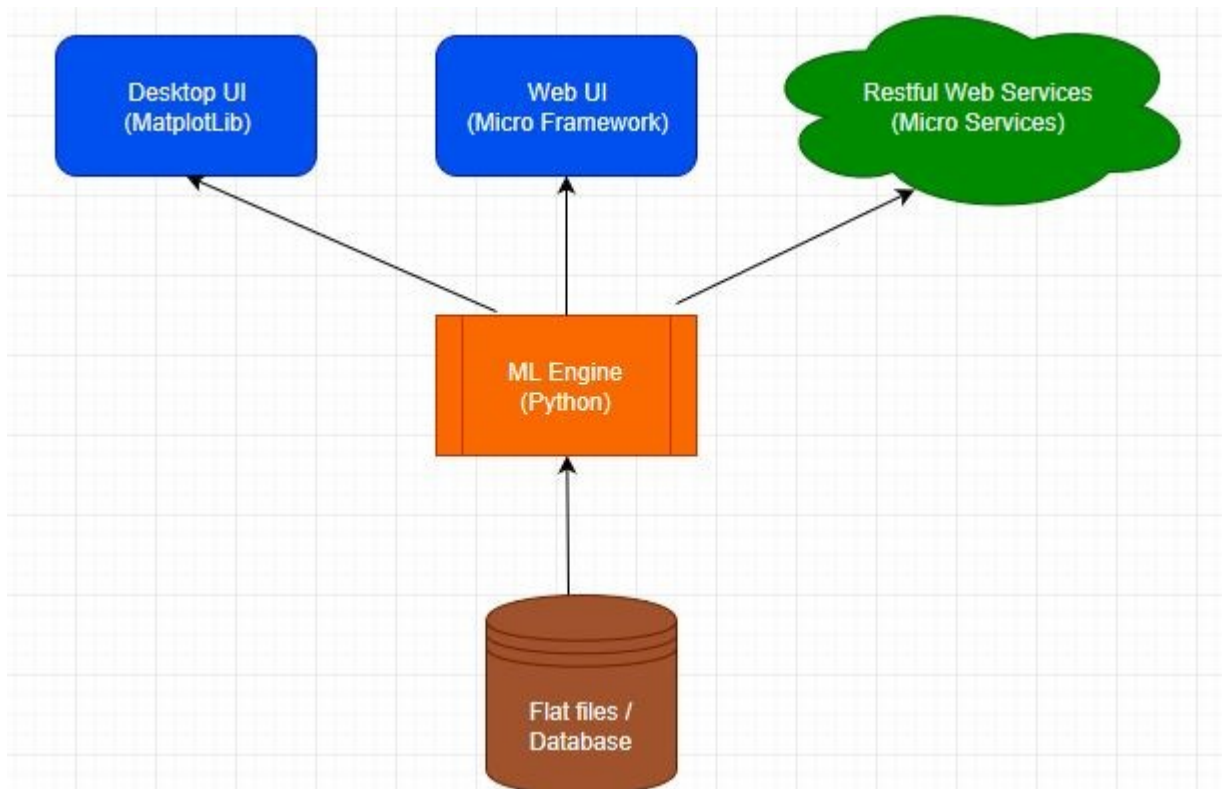


Figure 53:Architecture of the Integrated System

Main components of the system are persistence storage, processing layer and the interfaces. The processing layer is the machine learning engine which accesses data from persistence storage. The different forms of the interfaces interact with the engine to retrieve the predictions and then present them to the users or other systems.

Storage consists of a relational database and data files. Both training and test data are structured and stored on flat files. Data from those files are loaded into main memory and constructed as a multi-dimensional array using Python's Pandas library. The application reference data is stored and in a mysql database. Reference data is loaded and kept in-memory to enrich the data streams propagating through the machine learning models.

The core component of the system is the machine learning engine where main processing logic is encapsulated. The engine is designed and built as an independent module and is used by multiple other components like web services and user interfaces.

Two different types of user interfaces are available for users to interact with the application.

Desktop UI: Which can be run on client-side computer, consuming the service of ML Engine. It interacts with the ML engine to render the interactive UI. Python library *Matplotlib* (<https://matplotlib.org>) was the technology used in UI design.

Web UI: Which can be accessed from a remote web browser. It delegates user requests to the prediction engine and serves the response back. Python based microframework called *Flask* (<http://flask.pocoo.org/>) was used to leverage code to expose as a web application.

We adopted Microservice architecture which is also called Service Oriented Architecture (SOA) as our main architecture principle, so the application is by-design and provides the integration out of the box.

What that means is that it is much easier for external applications to integrate with MALEP and consume the services on offer.

The modern standard application integration method is API via the web services. To that end, a Restful web service was built on the calculation engine to provide the integration capability with other systems. It provides the API via HTTP using JSON as the data sharing file format. This enables the technology-agnostic integration. In other words, the capability to integrate with other systems that might use different technologies from Python. More details on how they were designed as a microservices environment is discussed in the next section.

7.2.2 Microservices for Integration

Given that we used Java based technology for the model-based web calculation tool, there was a challenge of how that tool could utilize the Python-based ML predictor (MALEP).

How we addressed this problem was by introducing a Component-Based software design. The core ML engine is segregated from the presentation component in a loosely coupled way, enabling it to be accessible and maintained independently.

ML engine and all its related components are designed in a pluggable way.

Figure 54 depicts how external applications would integrate with MALEP to consume the restful web services.

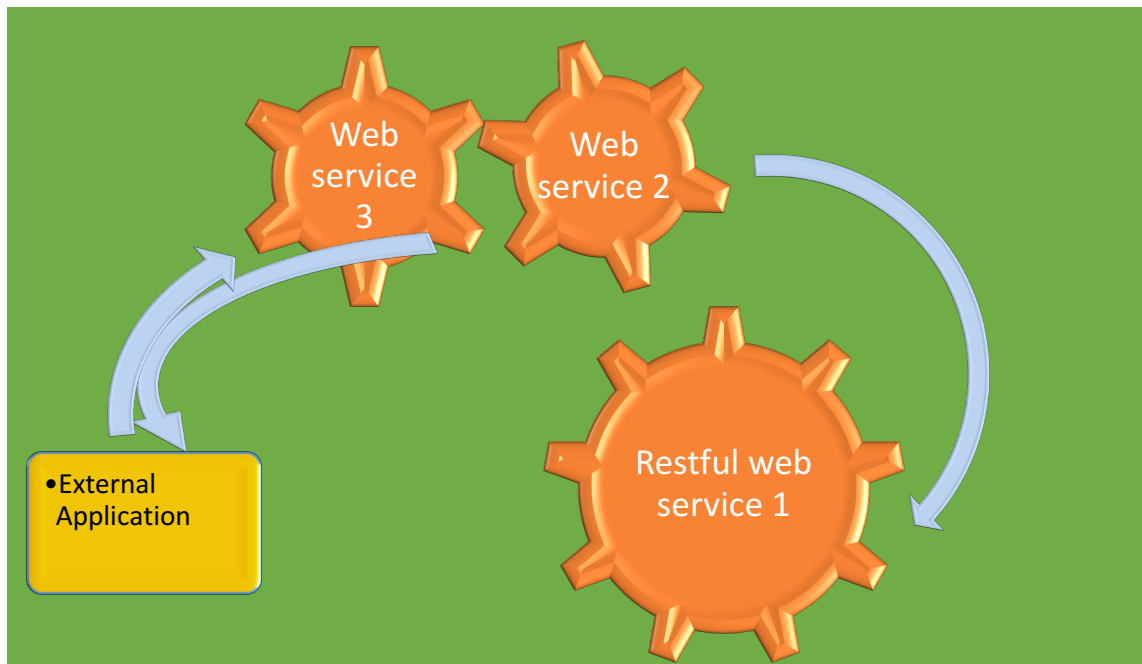


Figure 54:Application Integration

MALEP offers its prediction service as a web service API. Our own web tool we built for model-based calculations also falls into the external application category. It is integrated with MALEP web services and the service is accessed using JSON messages.

7.3 Technology

7.3.1 Python & EcoSystem

After a careful evaluation of the technology, we chose Python (www.python.org) as the primary programming language. Java was also a close candidate for ML due to its rich and mature eco-system and availability of open source libraries for ML.

However, Python was preferred mainly for its simple and easy to use untyped-data structures. That helps to parse and transform data easily which is a very powerful option to have.

Availability of excellent ML libraries for Python was also a major decisive factor. Examples of these open source supported libraries are *Pandas*, *numpy*, *keras* and *sklearn*. There are also useful data visualization libraries like *statsmodel* and all these are open source, hence freely available to use.

Python and relevant libraries Code samples are presented later in this chapter.

Below table lists all the open-source libraries used in MALEP:

Library	Purpose	Reference
<i>Pandas</i>	Data structures and data file handling	https://pandas.pydata.org
<i>Numpy</i>	Provides complex data structure like Multi-dimensional arrays and supports Scientific computing	https://numpy.org
<i>Sklearn</i>	Data migration and analysis	https://scikit-learn.org
<i>Statsmodel</i>	Statistical model estimation	https://www.statsmodels.org
<i>Keras</i>	Neural Network API which users TensorFlow internally. Supports both convolutional and recurrent networks	https://keras.io
<i>TensorFlow</i>	Core Neural network framework	https://www.tensorflow.org
<i>Matplotlib</i>	2D plotting library	https://matplotlib.org
<i>Seaborn</i>	Data visualization library built on matplotlib	https://seaborn.pydata.org

Table 10:Python Frameworks Used

7.3.2 Neural Network

As discussed in chapter 5, Neural Network resembles the human brain with connected neurons. The primary data structure of the neural network is the model. While Keras provides multiple models, the one we chose was **Sequential**. That is how the sequential model is constructed and assigned to a reference:

```
1. # Sequential Model construction
2. NN_model = Sequential()
```

The model consists of multiple layers. Firstly, the input layer with Dense of 128 is created and added to the model. That is where training data is fed into the model.

The Input Layer :

```
1. print(train.shape)
2. NN_model.add(Dense(128, kernel_initializer='normal', input_dim = train.shape[1]
, activation='relu'))
```

Three hidden layers are added to form the processing layers.

```
1. # The Hidden Layers :
2. NN_model.add(Dense(256, kernel_initializer='normal', activation='relu'))
3. NN_model.add(Dense(256, kernel_initializer='normal', activation='relu'))
4. NN_model.add(Dense(256, kernel_initializer='normal', activation='relu'))
```

And the output layer with a Dense of 1 is added to finish the model construction with the activation algorithm is linear.

```
1. # The Output Layer :
2. NN_model.add(Dense(1, kernel_initializer='normal', activation='linear'))
```

Eventually, the entire model is compiled by providing the matrices of absolute error and accuracy. Due to these matrices being added, it provides the ability of querying these matrices from the model.

Compile the network :

```
1. NN_model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mean_
absolute_error', 'accuracy'])
```


Figure 55 has the visual representation of the neural network built in the MALEP:

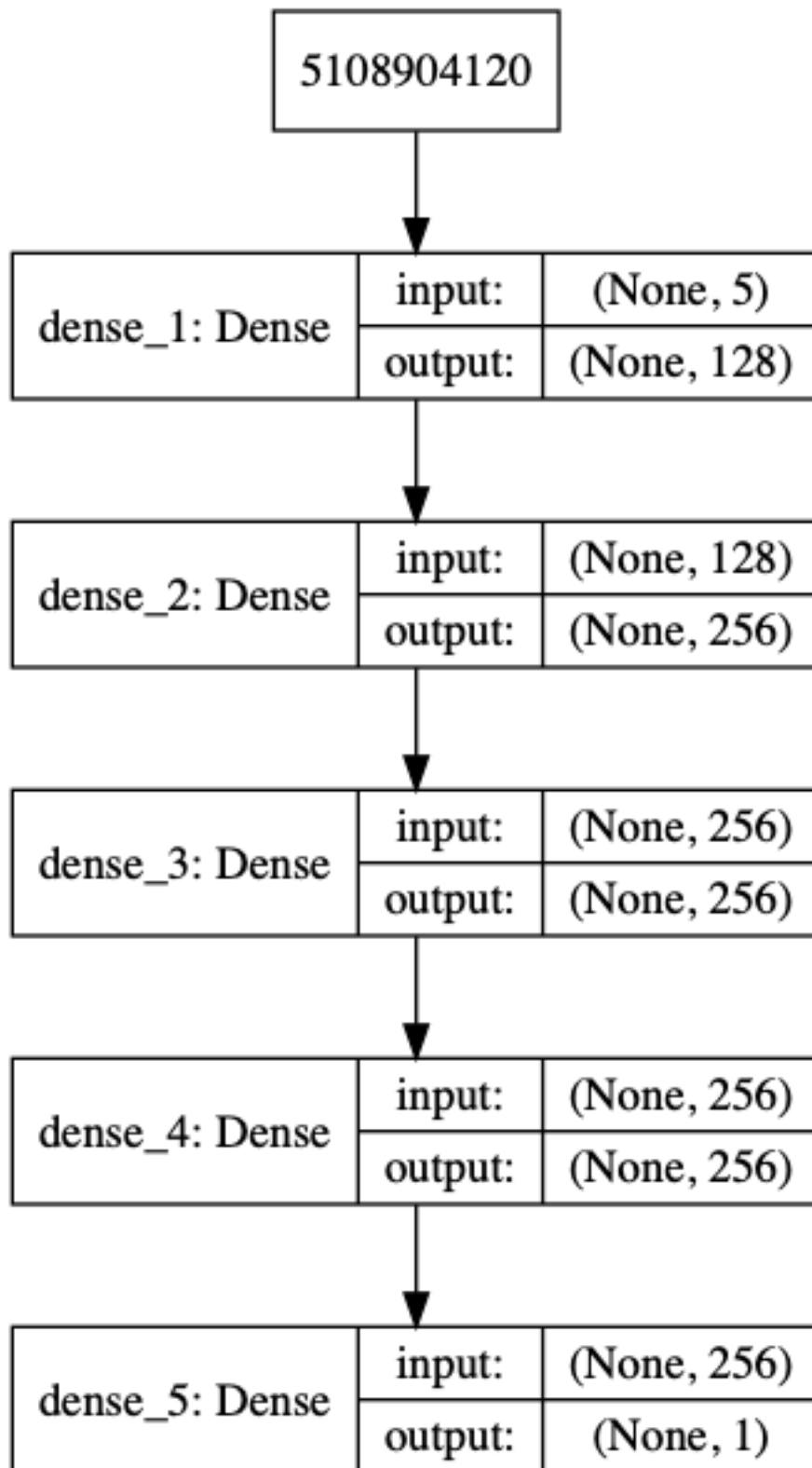


Figure 55:MALEP NN

7.3.3 Virtual Environment

For easy deployment and maintenance, all code is deployed in a Python Virtual Environment. A Python Virtual Environment is a self-contained directory tree which contains python and all dependencies pre-installed.

MALEP is shipped in a virtual environment and a user who downloads and executes it will not have to perform any sort of installations or dependency download. Everything is packaged in the virtual environment which means it is all good to execute the application straightaway.

A Virtual environment provides the portability to deploy the application into multiple environments with consistent behaviour across various platforms and operating systems. As an example, we could perform the isolated development testing in our local PCs and then promote the same code into the hosted Linux server for integration, by only changing the configurations.

Figure 56 below shows how the files are organized in the Python Virtual Environment.

```
.
├── bin
│   ├── activate
│   ├── activate.csh
│   ├── activate.fish
│   ├── easy_install
│   ├── easy_install-3.7
│   ├── pip
│   ├── pip3
│   ├── pip3.7
│   ├── python
│   ├── python3
│   └── python3.7
├── include
├── lib
│   └── python3.7
│       └── site-packages
│           ├── easy-install.pth
│           ├── pip-19.0.3-py3.7.egg
│           ├── setuptools-40.8.0-py3.7.egg
│           └── setuptools.pth
└── pyenvv.cfg
```

Figure 56 : MALEP Virtual Environment

7.4 Restful Webservices

REST (Representational State Transfer) is a lightweight but scalable web service architecture. The web services built using REST style are called Restful web services. The Restful web services we built are for external applications to integrate with MALEP to access the prediction service. They were designed by following the Micro Services and Micro Framework based architecture.

Security

The web service endpoint is only supported vi HTTP POST method. The requester needs to authenticate with the system by sending a pre-issued username and password as header attributes. MALEP first verifies the user credentials before serving the request.

Sample JSON Request

Paw (<https://paw.cloud>) was the HTTP utility we used to test the web services. The request is designed to contain independent attribute values with their names. Below is a screen for sample JSON.

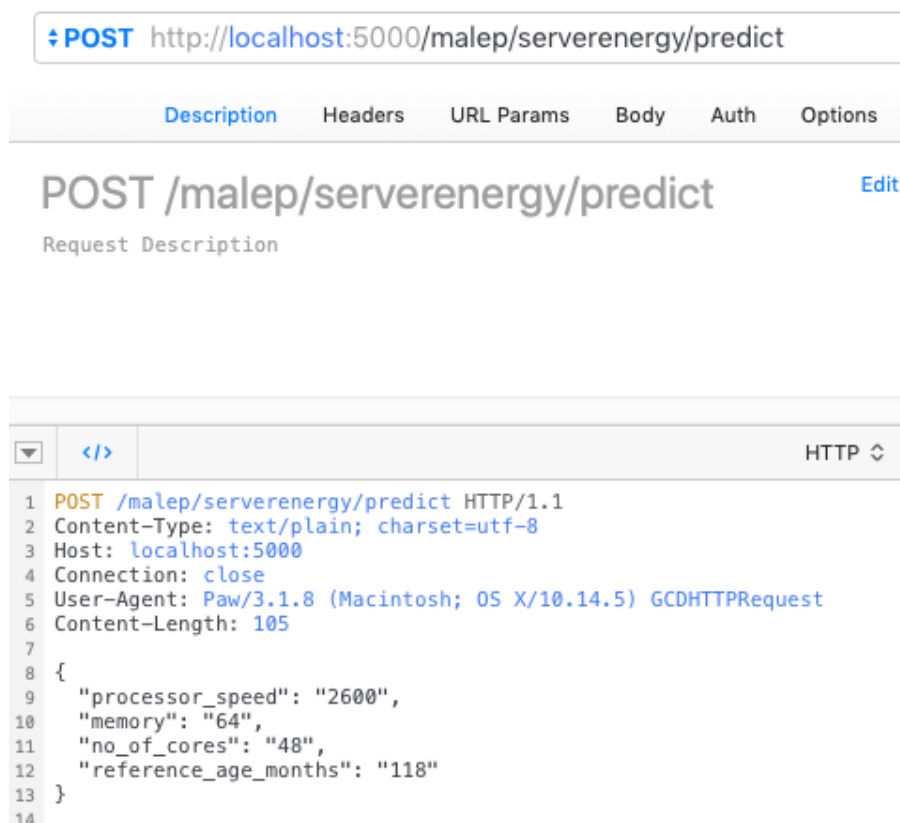


Figure 57 :JSON Request

Response with predicted values

Below is the response JSON received back from the restful web service with predicted energy consumption:

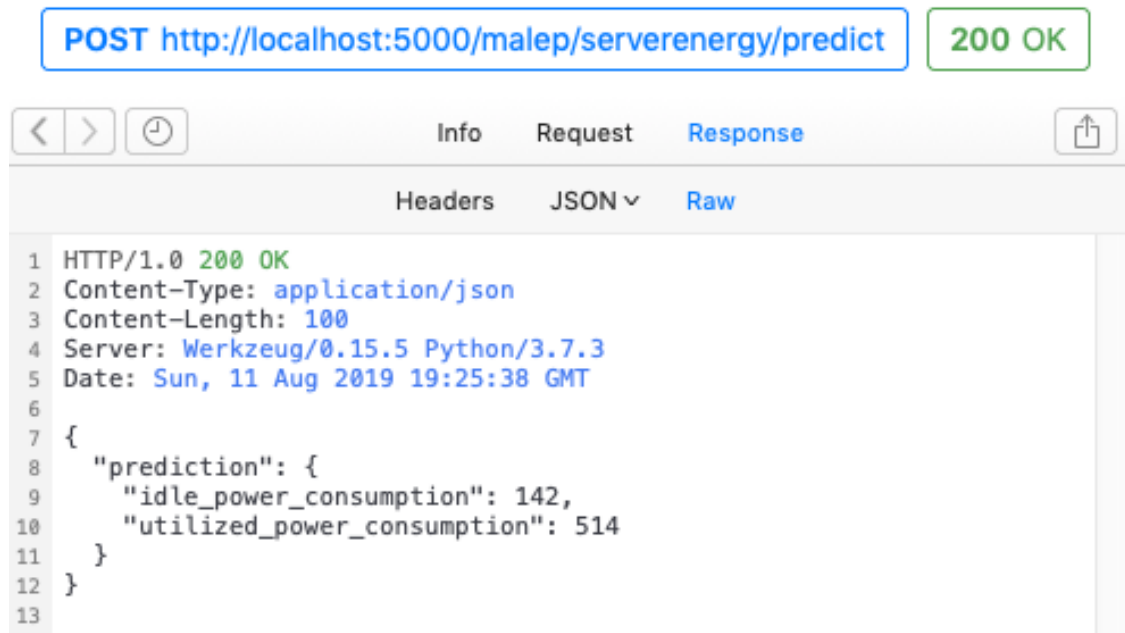


Figure 58 : JSON Response

Web Services Code

Below are some code samples responsible for serving Restful web service. The code first handles the JSON request and performs necessary validation. The request is then delegated to the ML engine to retrieve the predicted energy consumption. The result comes back in the form of a Python data object. It is then transformed into JSON before streaming back to the client.

```
1. @app.route('/malep/serverenergy/predict', methods=['POST'])
2. def get_predicted_energy():
3.     if not request.json :
4.         abort(400)
5.
6.     processor_speed = request.json['processor_speed']
7.     no_of_cores = request.json['no_of_cores']
8.     memory = request.json['memory']
9.     reference_age_months = request.json['reference_age_months']
10.
```

```

11.     predicted_utilized_energy = predict_utilized_power(processor_speed, no_of_
        cores, memory,
12.                                                                 refere
        nce_age_months)
13.
14.     predicted_idle_energy = predict_idle_power(processor_speed, no_of_cores, m
        emory,
15.                                                                 reference_age_months)
16.
17.
18.     predicted_consumption = {
19.         'utilized_power_consumption': json.dumps(predicted_utilized_energy),
20.         'idle_power_consumption': json.dumps(predicted_idle_energy)
21.     }
22.
23.     return jsonify({'prediction': predicted_consumption}), 201

```

7.5 Micro Framework and API development

Microframework helps to leverage existing functionality and expose it for clients via HTTP. It powers up the API development by facilitating to serve an underlying functionality as web services, most commonly Restful style.

The microframework we used in the application was *Flask* (<https://palletsprojects.com/p/flask/>). Flask is a Web Server Gateway Interface (WSGI) web framework.

We considered some other Python microframeworks like Bottle (<https://bottlepy.org>), but eventually, Flask was the clear winner.

The reason for choosing it over other possible candidates was due to its lightweight architecture and the ability to build products relatively easily and quickly. Another reason was its robust scalability, which helps to scale up the service to meet high demand automatically in production environments. The ability to scale up and down automatically to meet the changes in the demand is called elasticity in the industry.

The figure below represents the API architecture we built using Bottle to expose energy consumption predictors as web services. The JSON requests coming in are intercepted and validated before routed to the appropriate service component. The individual predictor within the MALEP engine uses the incoming server attributes and sends back the predicted values to the API layer. The values are then propagated back to the client in the response. The processing transaction details are logged for troubleshooting and auditing purposes by the logging module:

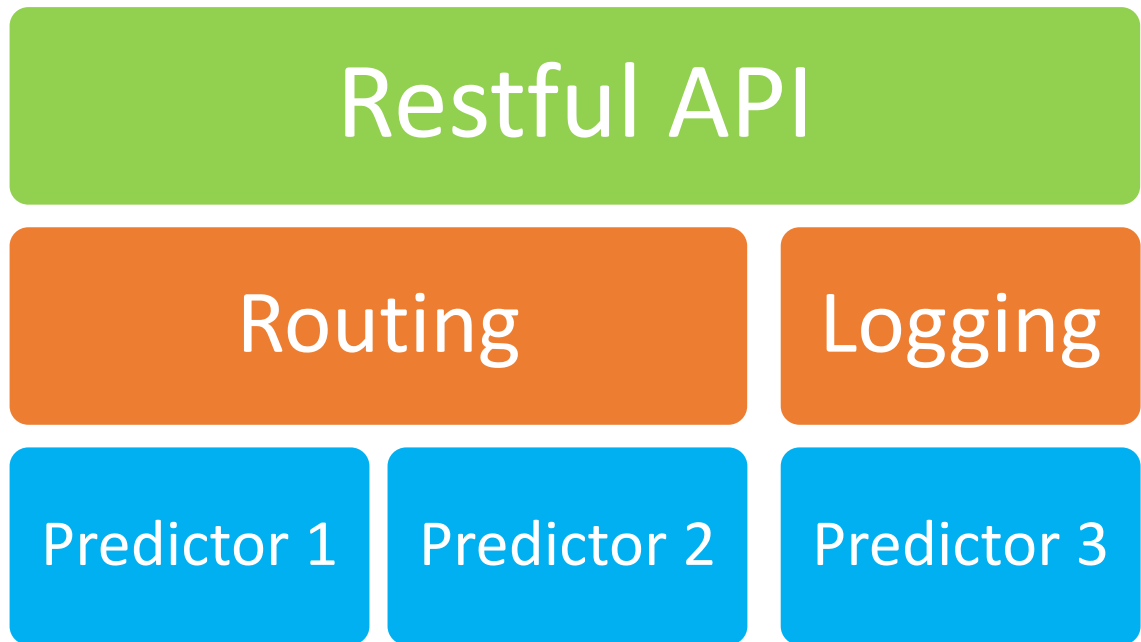


Figure 59:MALEP Microframework

7.6 Challenges faced

One major challenge was how to execute the application smoothly and quickly for the potential users once downloaded. This is due to the fact that application requires a number of dependencies which need to be downloaded and built first. This can be even more complex process as users may use them on different operating. Systems like Window, Mac and Linux. This challenge was successfully overcome by using the Python Virtual environment (7.3.3) as it helps to bundle the application and all its required dependencies in a container. This container helps the application to work like in an isolated environment, regardless of which operating system it runs on or the underlying hardware specifications.

7.7 Summary

We explained the key design and architecture principles of the MALEP with diagramme. Comprehensive details about technology evaluation process and the decisions is also present in this chapter. How the modern technology was utilized to implement the tool as per the main design is explained with sample code.

Part IV

Evaluation

“Many of life’s failures are people who did not realize how close they were to success when they gave up.”

-Thomas A. Edison

CHAPTER XIII

EVALUATION OF MALEP: REGRESSION MODELS

This chapter is about how the ML regression and deep learning algorithms were tested and evaluated using the predictor tool. We used a wide variety of benchmarks and scenarios, outlined throughout this chapter.

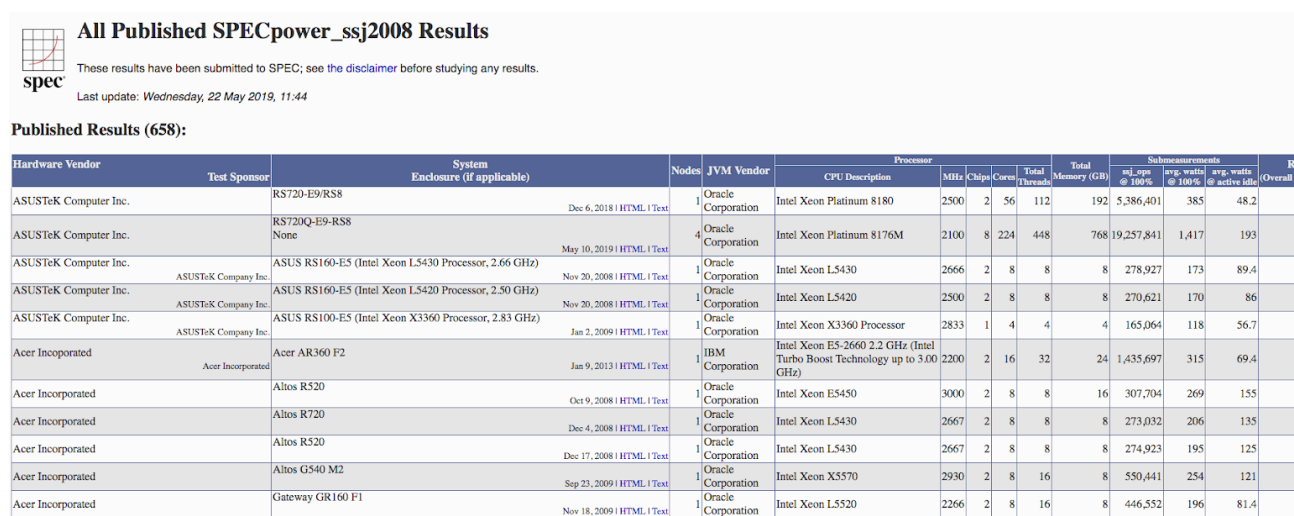
8.1 Test Data

8.1.1 Obtained data set

Standard Performance Evaluation Corporation (SPEC) publishes a list of server specifications with associated attributes and the energy consumption figures for each wide variety of specification. We chose their data due to the quality and the reliability. The fact that the data is available in the public domain is also another reason behind that decision.

A selected data set consists of several attributes for server specifications along with utilized and idle energy consumption in watts.

Figure 60 is a sample set of data in the original format:



All Published SPECpower_ssj2008 Results
 These results have been submitted to SPEC; see the disclaimer before studying any results.
 Last update: Wednesday, 22 May 2019, 11:44

Published Results (658):

Hardware Vendor	Test Sponsor	System Enclosure (if applicable)	Nodes	JVM Vendor	Processor				Total Memory (GB)	Submeasurements			Overall
					CPU Description	MHz	Chips	Cores		Total Threads	sj_ops @ 100%	avg. watts @ 100%	
ASUSTeK Computer Inc.		RS720-E9/RS8 Dec 6, 2018 HTML Text	1	Oracle Corporation	Intel Xeon Platinum 8180	2500	2	56	112	192	5,386,401	385	48.2
ASUSTeK Computer Inc.		RS720Q-E9-RS8 None May 10, 2019 HTML Text	4	Oracle Corporation	Intel Xeon Platinum 8176M	2100	8	224	448	768	19,257,841	1,417	193
ASUSTeK Computer Inc.	ASUSTeK Company Inc.	ASUS RS160-E5 (Intel Xeon L5430 Processor, 2.66 GHz) Nov 20, 2008 HTML Text	1	Oracle Corporation	Intel Xeon L5430	2666	2	8	8	8	278,927	173	89.4
ASUSTeK Computer Inc.	ASUSTeK Company Inc.	ASUS RS160-E5 (Intel Xeon L5420 Processor, 2.50 GHz) Nov 20, 2008 HTML Text	1	Oracle Corporation	Intel Xeon L5420	2500	2	8	8	8	270,621	170	86
ASUSTeK Computer Inc.	ASUSTeK Company Inc.	ASUS RS100-E5 (Intel Xeon X3360 Processor, 2.83 GHz) Jan 2, 2009 HTML Text	1	Oracle Corporation	Intel Xeon X3360 Processor	2833	1	4	4	4	165,064	118	56.7
Acer Incorporated	Acer Incorporated	Acer AR360 F2 Jan 9, 2013 HTML Text	1	IBM Corporation	Intel Xeon E5-2660 2.2 GHz (Intel Turbo Boost Technology up to 3.00 GHz)	2200	2	16	32	24	1,435,697	315	69.4
Acer Incorporated		Altos R520 Oct 9, 2008 HTML Text	1	Oracle Corporation	Intel Xeon E5450	3000	2	8	8	16	307,704	269	155
Acer Incorporated		Altos R720 Dec 4, 2008 HTML Text	1	Oracle Corporation	Intel Xeon L5430	2667	2	8	8	8	273,032	206	135
Acer Incorporated		Altos R520 Dec 17, 2008 HTML Text	1	Oracle Corporation	Intel Xeon L5430	2667	2	8	8	8	274,923	195	125
Acer Incorporated		Altos G540 M2 Sep 23, 2009 HTML Text	1	Oracle Corporation	Intel Xeon X5570	2930	2	8	16	8	550,441	254	121
Acer Incorporated		Gateway GR160 F1 Nov 18, 2009 HTML Text	1	Oracle Corporation	Intel Xeon L5520	2266	2	8	16	8	446,552	196	81.4

Figure 60: A sample server specification with energy consumption data from SPEC

8.1.2 Data Cleansing and Feature Selection

The raw data from SPEC is not in a suitable format to train the ML algorithms. This is due to the structure of the data and the non-standard nature of it. Also, some of the attributes are not directly relevant to the server energy consumption. Java Virtual Machine vendor name and the company names who sponsored the test provision are such attributes. So, we only selected the relevant attributes and cleansed the raw data and then structure it in a format suitable for ML model evaluation.

8.1.2.1 Cleansed Training Data

A few rows from the cleansed training data is listed in table 11:

Fully Utilized Power	Active Idle Power	Number of Cores	Processor Speed	Memory	Reference Age in Months
118	56.7	4	2833	4	63
170	86	8	2500	8	69
385	48.2	56	2500	192	188
1417	193	224	2100	768	192
315	69.4	16	2200	24	112
125	35.4	4	2933	8	86
58	21.5	4	2400	8	104
267	76.9	12	2933	12	89
124	39.6	4	2933	8	86
272	73.6	12	2933	12	89

Table 11 : Cleansed training data sample

The training data set was then converted into a tabular format for easy access by the computer programmes. The tabular format also works well in data transformation through the application code. The chosen format was a comma-separated value (CSV) file.

Some of the training data in a tabular format are shown in figure 61:

average_watts_fully_utilized,	average_watts_active_idle,	no_of_cores,	processor_speed,	memory,	reference_age_months
118,	56.7,	4,	2833,	4,	63
170,	86,	8,	2500,	8,	69
385,	48.2,	56,	2500,	192,	188
1417,	193,	224,	2100,	768,	192
315,	69.4,	16,	2200,	24,	112
125,	35.4,	4,	2933,	8,	86
58,	21.5,	4,	2400,	8,	104
267,	76.9,	12,	2933,	12,	89
124,	39.6,	4,	2933,	8,	86
272,	73.6,	12,	2933,	12,	89

Figure 61 :Cleansed training data sample in tabular format

8.1.3 Test Data Analysis

The statistical analysis on the chosen test data is presented below. The main objective behind the analysis was to examine if the quality and the distribution of data is good enough for evaluation purposes.

8.1.3.1 Distribution of Data

Distribution of each training data feature is visualized below. X-axis is the value of the given feature and the Y-axis is the number of test data items.

Fully utilized power

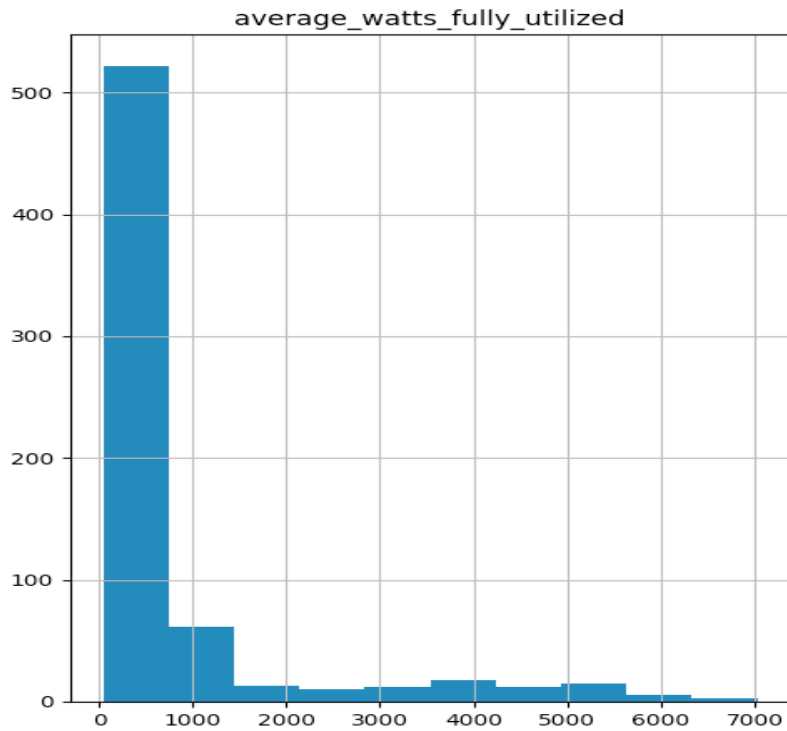


Figure 62 : Distribution of fully utilized power

Idle Power Consumption

Figure 63 is the distribution of the idle power consumption in the test data set:

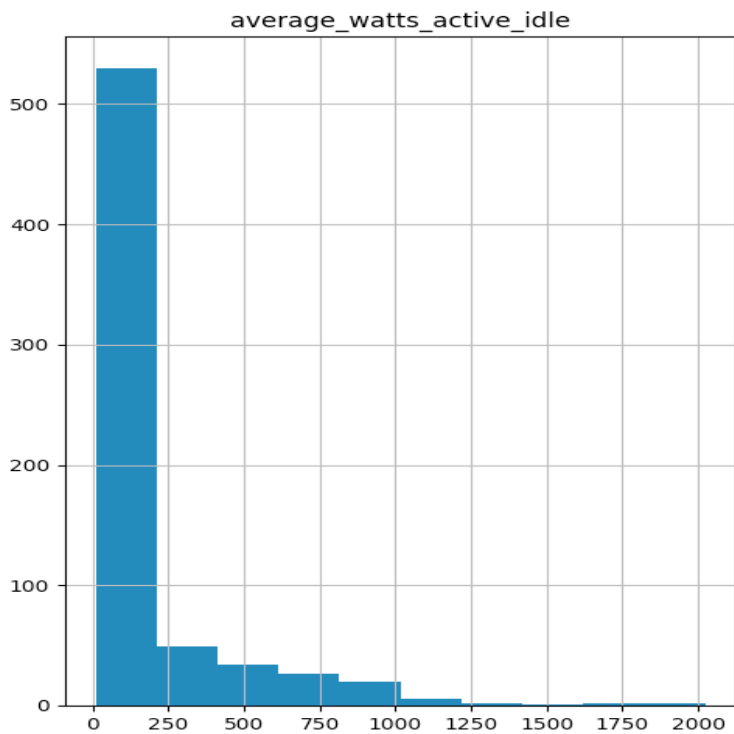


Figure 63: Distribution of idle power

Processor speed

Figure 64 is the visualised distribution of processor speed:

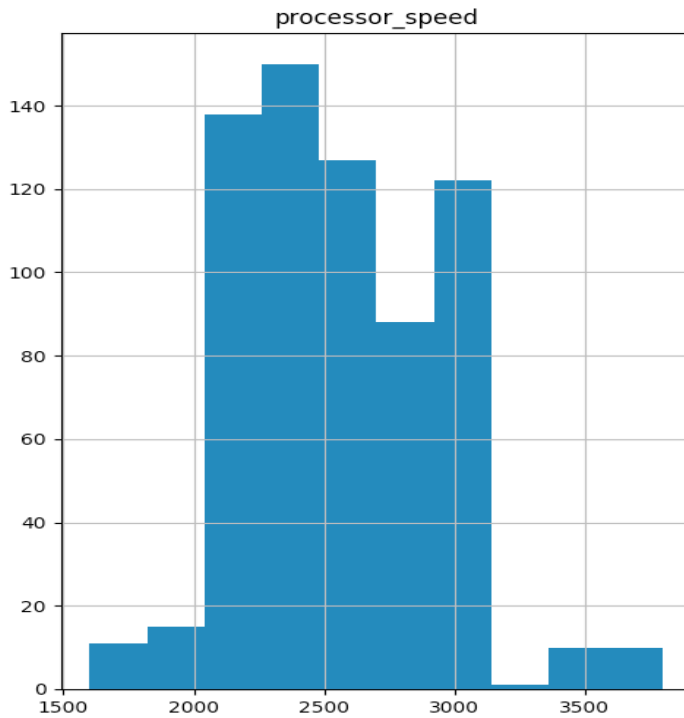


Figure 64: Distribution of Processor Speed

As we can see in the graph, most of the servers built with CPU speed between 2000Ghz and 3000Ghzs

Memory

Figure 65 is the distribution of memory:

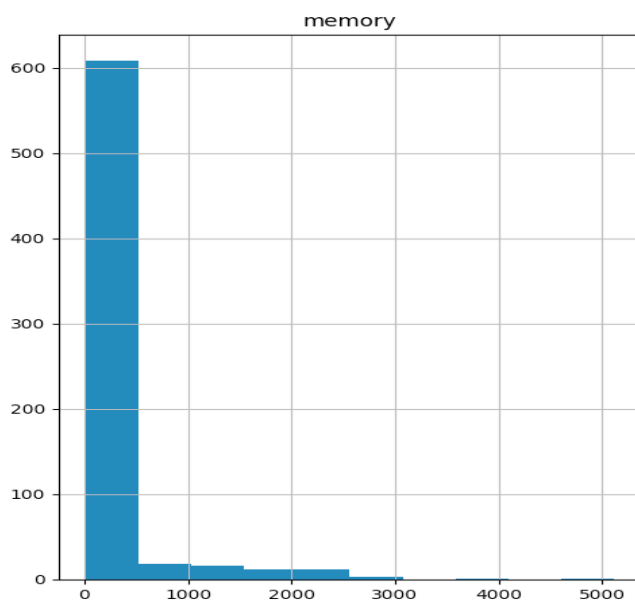


Figure 65: Distribution of memory

Number of processor cores

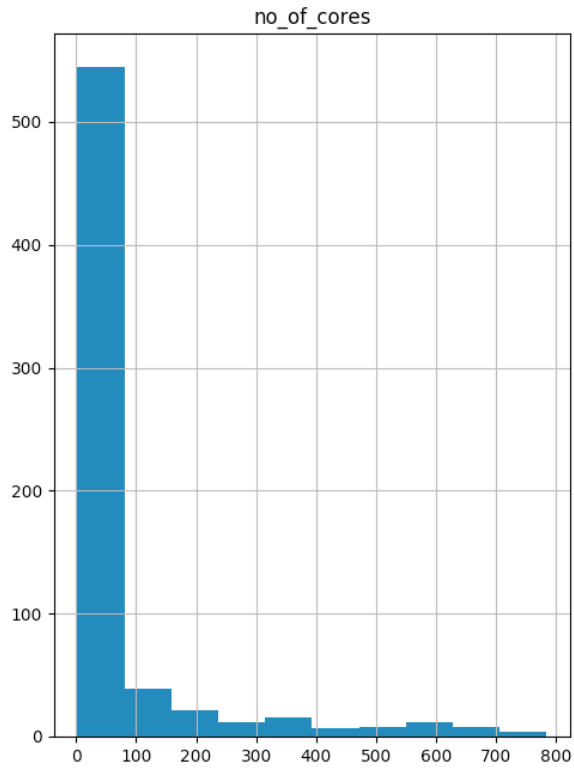


Figure 66: Distribution of number of cores

Reference Age

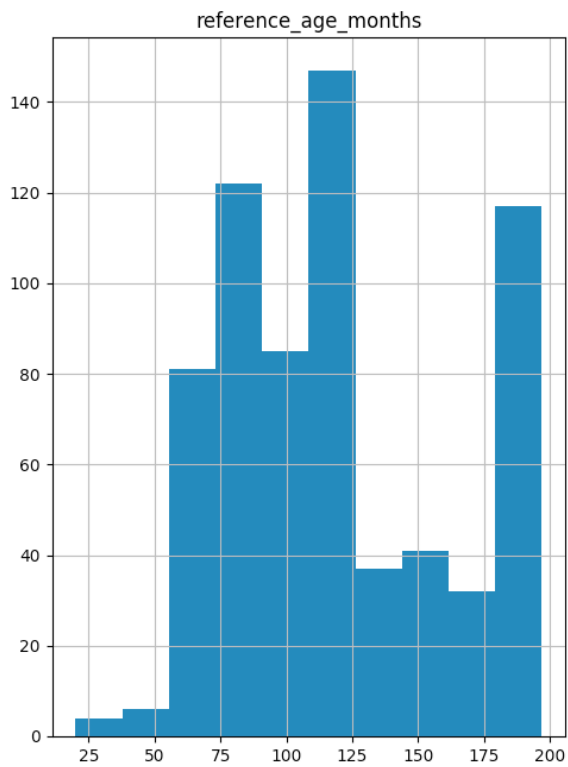


Figure 67: Distribution of reference age

8.1.4 Statistical Details of Full Data Set

Statistical details of the test data are depicted in table 12:

Attribute	Minimum	Max	Mean	Standard Deviation
Fully Utilized Power Consumption	44.70	7031	823.22	1334.47
Idle power consumption	9.33	2024	188.53	267.94
Processor Speed	1600	3800	2533.09	361.50
Memory	4	5120	219	527.74
Number of Cores	2	784	77.22	145.44
Threads per Core	1	4	1.77	0.427
Reference Age	20	197	119.22	43.52

Table 12 : Summary of Full data set

8.1.5 Test Data Utilization Strategy

We split the test data set of 673 records into two different sets. The first set of 449 is for training the machine learning models and the rest of 224 data elements are used to test the trained models.



Figure 68: Test Data Strategy

1. Training Data set – 449 elements

The data set used for training the model.

2. Evaluation Data set – 224 elements

The data set of testing the trained models.

8.1.6 Statistical Analysis of Training Data Set

Table 13 is the statistical analysis on the training data set:

Attribute	Minimum	Max	Mean	Standard Deviation
Fully Utilized Power Consumption	44.7	7031	794.06	1270.79
Idle Power Consumption	10	2024	185.42	262.14
Processor Speed	1600	3800	2531.32	361.01
Memory	4	5120	209.62	509.85
Number of Cores	2	784	73.75	136.28
Reference Age	20	197	119.25	43.80

Table 13 : Training Data Analysis

8.1.7 Statistical Analysis of Test Data Set

Table 14 is the statistical analysis on the data set which was used to test the models:

Attribute	Minimum	Max	Mean	Standard Deviation
Fully Utilized Power consumption	51.8	6532	990.02	1650.09
Idle power consumption	9.33	1842	206.35	299.83
Processor Speed	1800	3700	2543.25	365.95
Memory	4	2688	272.68	620.49
Number of Cores	2	784	97.14	189.15
Reference age	40	197	119.11	42.11

Table 14: Test Data Analysis

8.2 Regression Model Evaluation

We executed the regression models using separate training and testing dataset by splitting the dataset into 2:1 ratio with training and testing set respectively as below:

Size of training dataset	Size of testing dataset
449	224

Table 15: Test Data Summary

8.2.1 Model Evaluation Criteria

The process we followed for model evaluation and benchmarking are shown below:

1. Model is trained using the training data set.
2. For each test data item, use a chosen one or more independent attributes as input parameters.
3. Predict utilized and idle energy consumption by executing the already trained model for each test data.
4. The predicted value is compared with the actual dependent variable (e.g. fully utilized consumption), and the error percentage is calculated using the below formula.

$$\text{Error percentage} = \left(\frac{\text{abs}(\text{predicted Value} - \text{Actual Value})}{\text{Actual Value}} \right) * 100$$

Equation 13 : Error Percentage Calculation

5. The error percentage for each test data row is accumulated and then the average error percentage is calculated.
6. The average error percentage is used to benchmark each model.

8.2.2 Evaluation of Fully Utilized Energy Consumption

We first worked on predicting the server power consumption when it is fully utilized. A few scenarios were picked by changing the independent input variables of the model. The dependent variable for all scenarios was fully utilized power consumption. The result analysis of each scenario with the test results are discussed under each scenario.

The benchmark of comparing each scenario is the average error percentage. All six independent parameters were used as independent variables in the scenario 6, expecting it to provide the best model accuracy. On the other hand, scenario 3 evaluates the model when only reference age is used as input. In other scenarios, different number of parameters as well different parameter combinations were evaluated. In reality also, it is very likely that all parameter values are not available for the data centre operators to use

in the prediction models. Benchmarking the performance of such models which depend on the server attribute values available, provides insights on the impact of different variables.

Scenario 1

Independent variable	Dependent Variable
Processor Speed Number of Cores	Fully Utilized Power Consumption

Results

After executing the model, the intercept and coefficients for each input parameter and the average error percentage are listed in below table.

Independent variable	Coefficient	Intercept (Constant)	Average Error %
Processor Speed	0.10	-119.59	31.36 %
Number of Cores	7.69		

Machine Learning Model

As a result of the training phase, the model for predictions is the one below:

<p>Utilized Consumption</p> $= -119.59 + 0.10 * \text{Processor Speed} + 7.69 * \text{No of Cores}$
--

We then executed the algorithm for the test data set. The first 10 predicted results are listed in the below table. It contains the input values, the actual and predicted consumption, along with error percentage.

Processor speed	Cores	Actual consumption	Predicted consumption	prediction error %
-----------------	-------	--------------------	-----------------------	--------------------

2666	8	173	221.59	28.09
2933	12	259	284.27	9.76
2500	12	307	240.44	21.67
2600	48	511	571.49	11.83
2400	64	713	693.87	2.68
2600	48	559	571.49	2.23
2500	12	327	240.44	26.47
2200	16	255	245.73	3.63
3067	12	264	297.84	12.82
3067	48	1025	618.75	39.63

Predicted values against actual consumption are visualized in the figure 69 and 70:

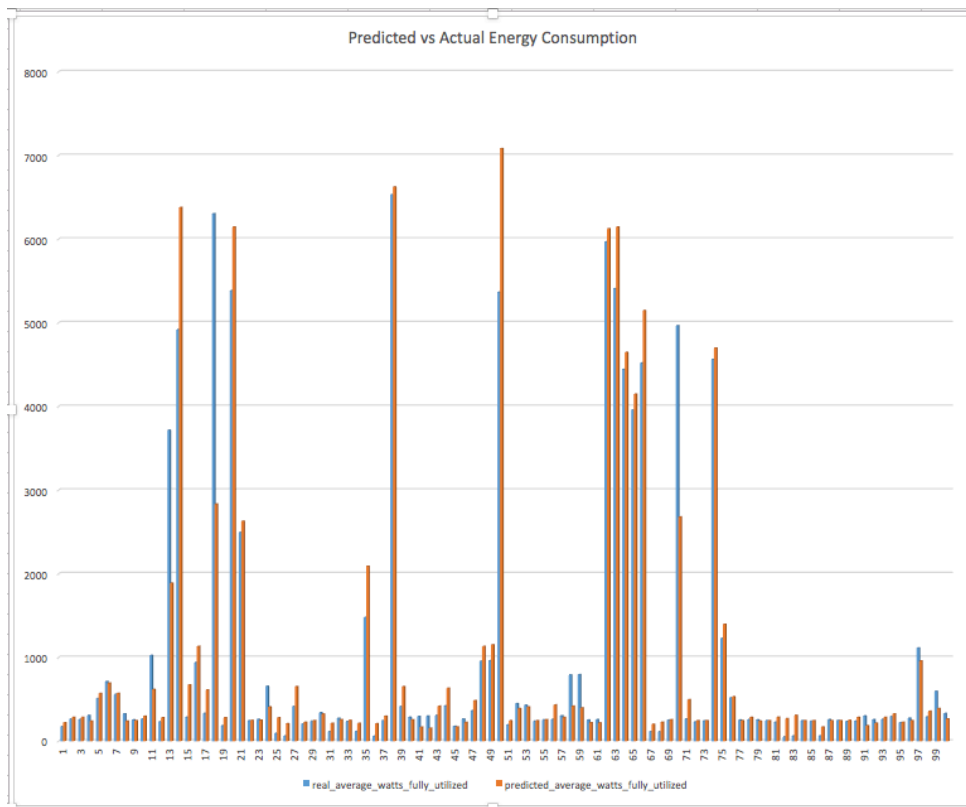


Figure 69: Predicted Vs Actual

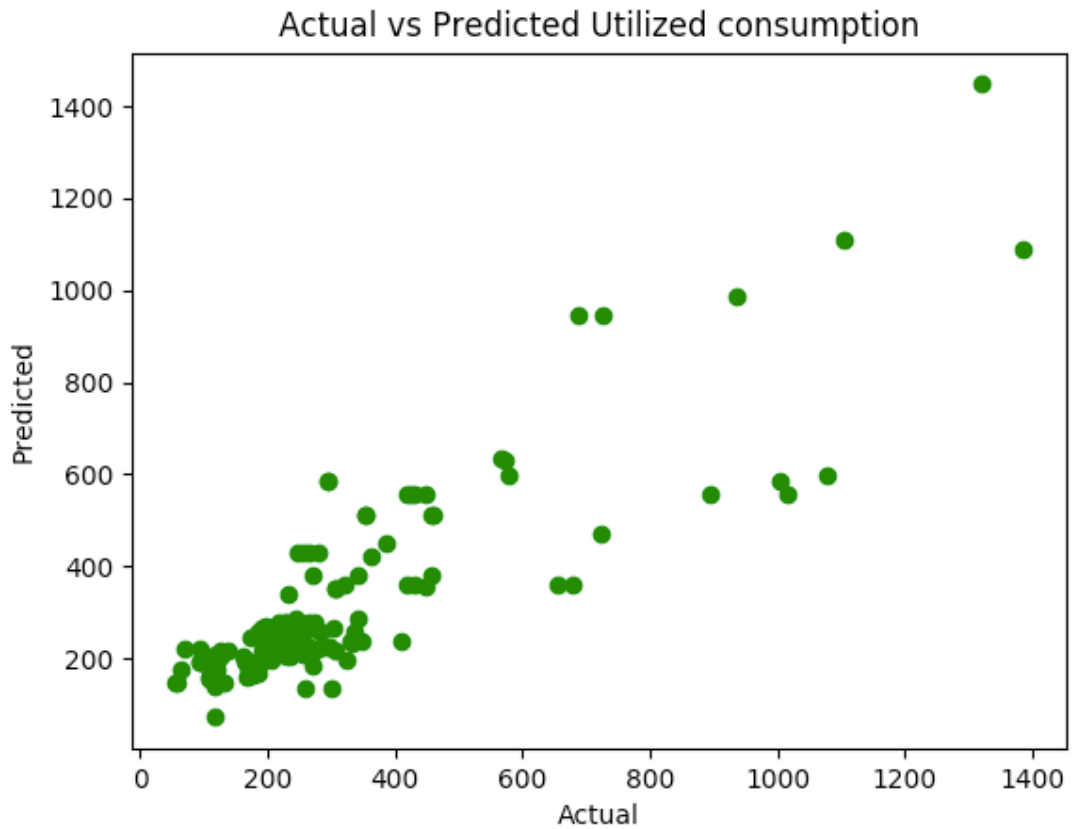


Figure 70: Actual Vs Predicted

Scenario 2

In the scenario 2, we used four input parameters in the evaluation as listed in the table below:

Independent variable	Dependent Variable
Processor Speed Number of Cores Memory Reference Age (months)	Fully Utilized Power Consumption

The same as in scenario 1, after the model executed with training data, the intercept for the model and coefficient for each input parameters were derived as shown in the table below:

Independent variable	Coefficient	Intercept (Constant)	Average Error %
Processor Speed	0.10	176.36	30.68%
Number of Cores	10.03		

Memory	-1.24		
Reference Age	-2.33		

Machine Learning Model

Based on the trained dataset, the model used for predictions is shown below:

<p>Average watts when fully utilized</p> $= 176.36 + 0.10 * \textit{Processor Speed} + 10.03$ $* \textit{Cores} + -1.24 * \textit{Memory} + -2.33$ $* \textit{Reference Age}$
--

The first 10 results rows of the results are shown in the table below:

Processor Speed	Number of Cores	Memory	Reference Age	Real Utilized Power	Predicted Utilized Power	Prediction Error Percentage
2666	8	8	69	173	228.88	32.30
2933	12	12	89	265	342.06	29.08
2933	12	12	89	259	342.06	32.07
2500	12	128	112	307	235.32	23.34
2600	48	64	118	511	552.60	8.14
2400	64	256	113	713	725.03	1.68
2600	48	64	108	559	594.46	6.34
2500	12	128	112	327	235.32	28.03
2200	16	24	124	255	209.54	17.82
3067	12	12	98	264	308.84	16.98

Predicted Vs Actual Consumption

A plotted graph of actual vs predicted utilised power consumption is shown below:

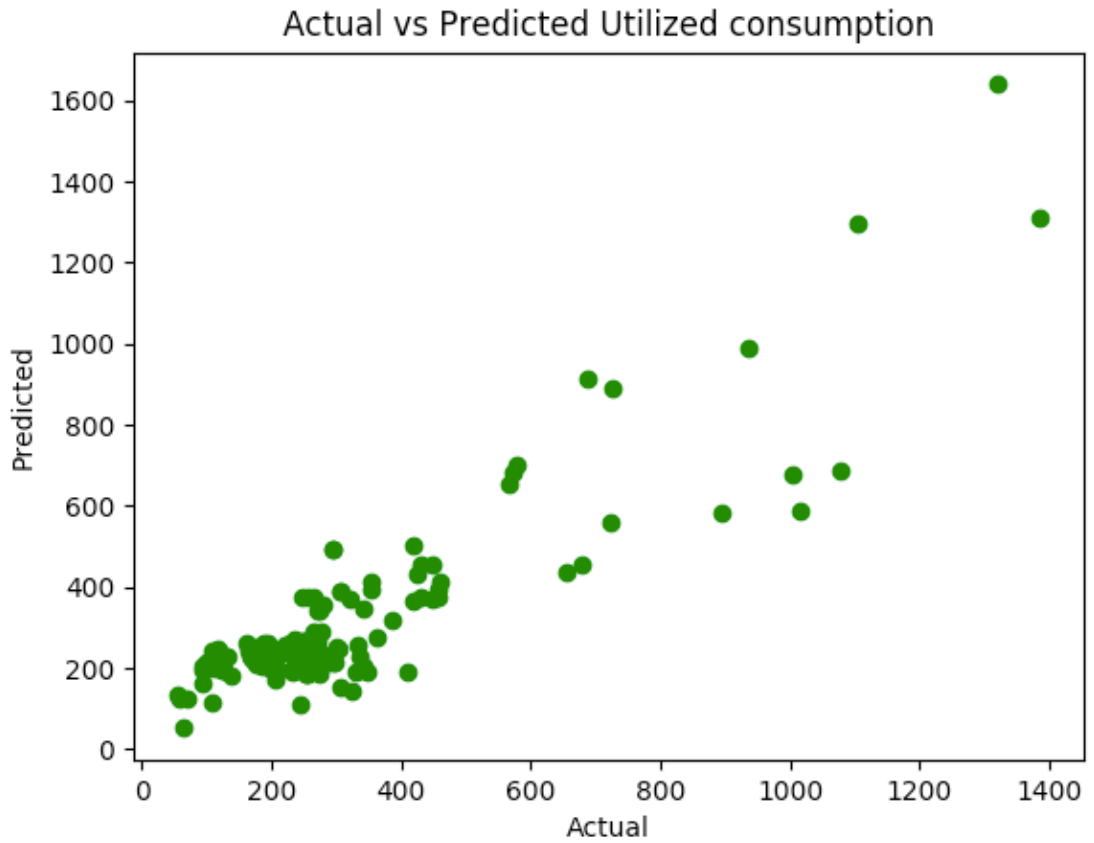


Figure 71: Predicted Vs Actual -Scenario 2

Actual Vs Predicted in a line chart is presented below:

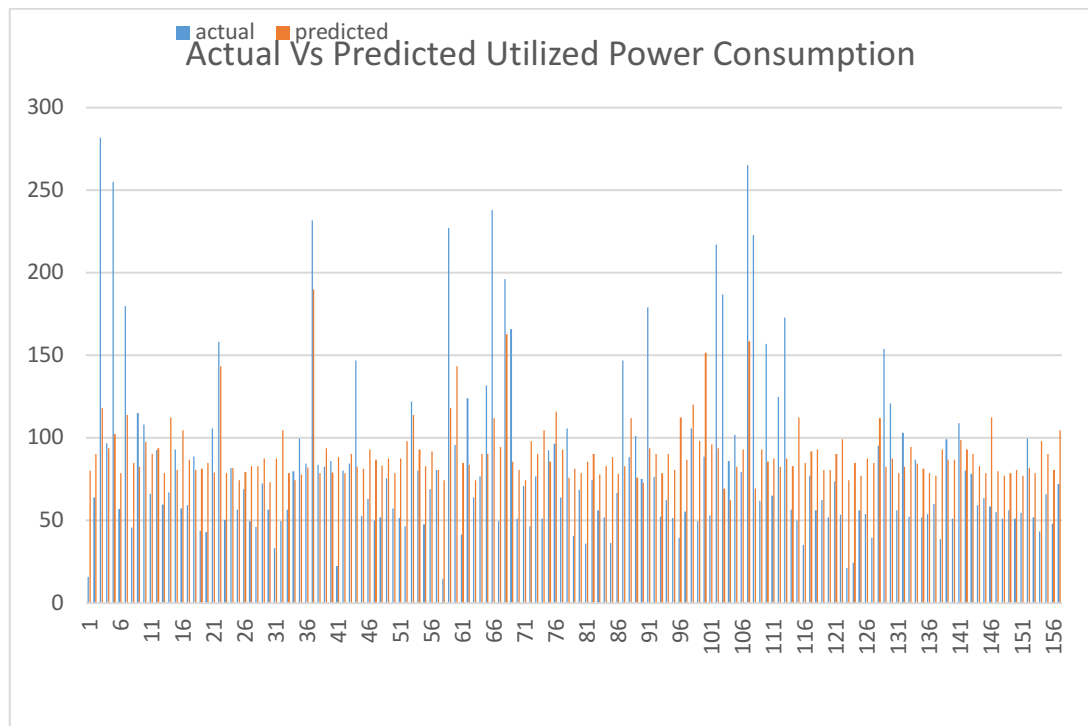


Figure 72: Actual Vs Predicted

Scenario 3

In this scenario, only one independent variable which is the reference age is used:

Independent variable	Dependent Variable
Reference Age (months)	Fully Utilized Power

And the results are in the table below:

Independent variable	Coefficient	Intercept (Constant)	Average Error %
Reference Age	1.27	166.98	54.93%

We see the error percentage has gone up when only reference age is used to feed the algorithms. This is an interesting observation as it's natural assume that that reference age more related to the power consumption. However, the reason why it is not highly related is because the age of a server doesn't carry a lot of information. Other factors like processor speed or memory have high impact on the energy consumption but do not necessarily rely on the age.

Machine Learning Model

Based on the trained dataset, the model used for predictions is shown below:

$$\text{utilized Consumption} = 166.98 + 1.27 * \text{Reference Age}$$

Detailed Prediction Model Evaluation Results (First 10 results only):

Reference Age	Real Fully Utilized Power	Predicted Fully Utilized Power	Prediction Error Percentage
69	173	240.51	39.03

89	265	460.83	73.89
89	259	460.83	77.92
112	307	714.20	132.64
118	511	780.29	52.69
113	713	725.22	1.71
108	559	670.14	19.88
112	327	714.20	118.41
124	255	846.39	231.92
98	264	559.98	112.11

Figure 73 shows the actual vs predicted consumption:

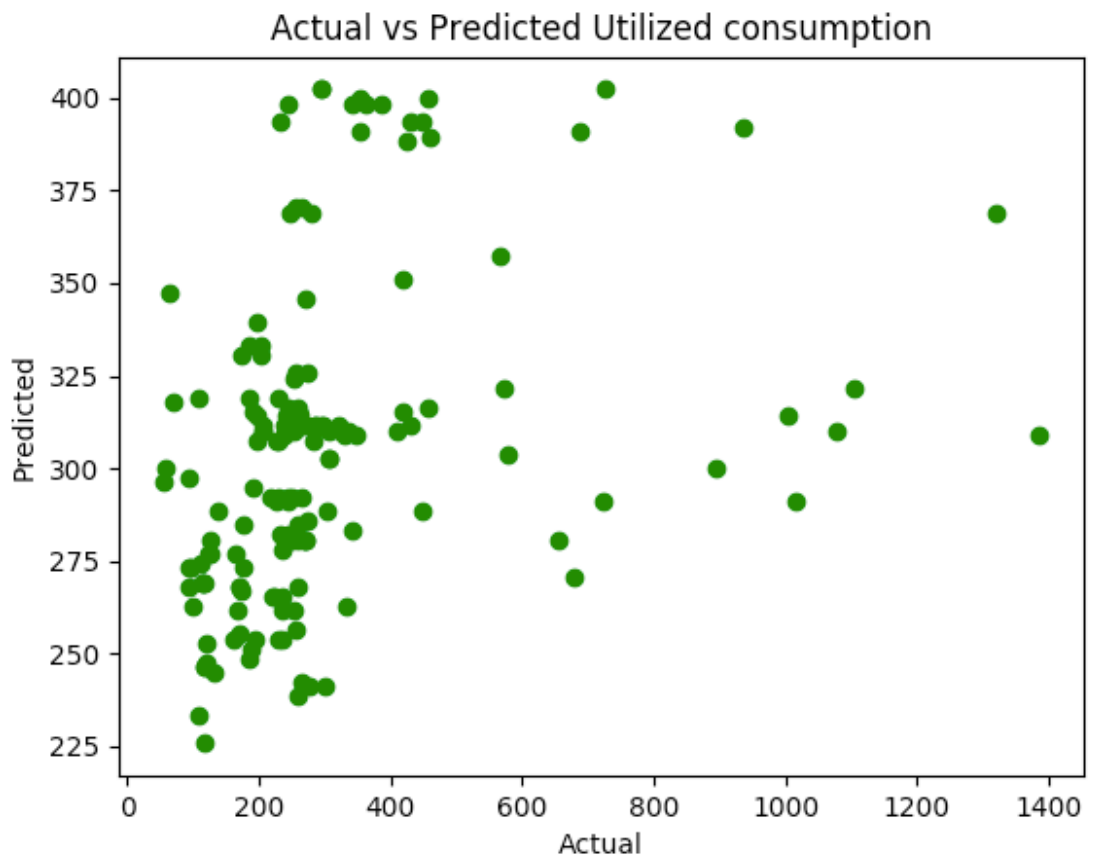


Figure 73: Predicted Vs Actual - Scenario 3

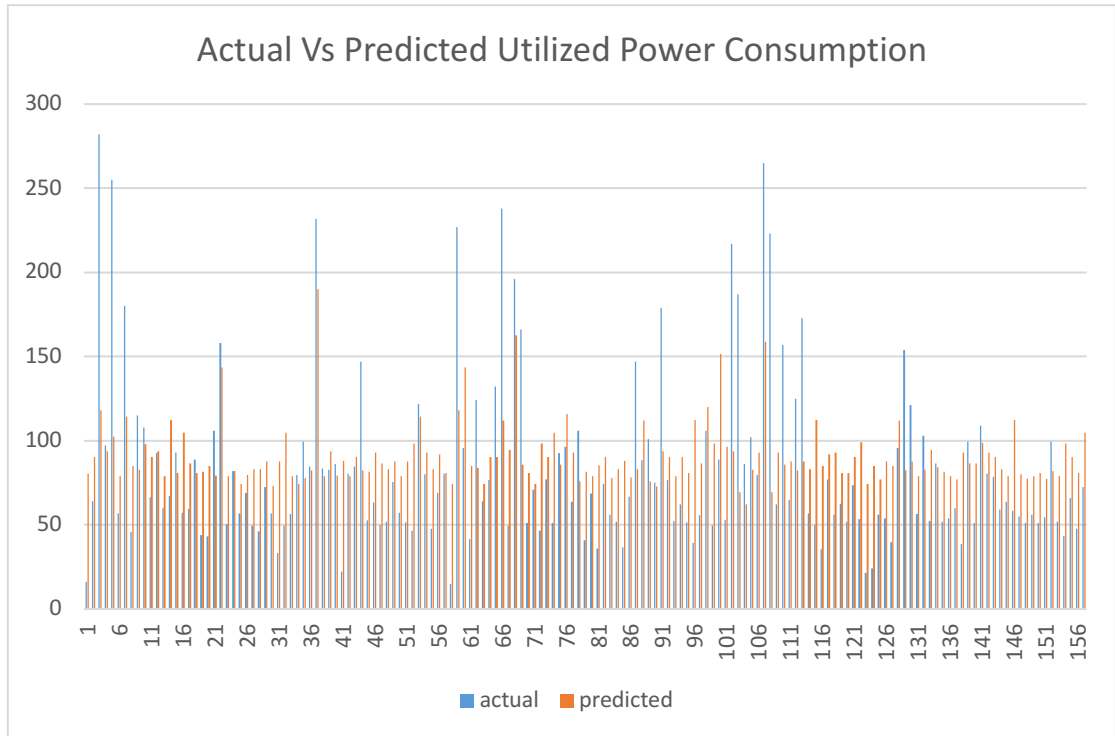


Figure 74: Actual Vs Predicted

Scenario 4

Here, three attributes were used as shown below:

Independent variable	Dependent Variable
Processor Speed	Fully Utilized Power Consumption
Number of Cores	
Reference Age (months)	

The results in the table below:

Independent variable	Coefficient	Intercept (Constant)	Average Error %
Processor Speed	0.04	241.08	30.46%
Cores	9.54		
Reference Age	-2.50		

Machine Learning Model

Based on the trained dataset, the model used for predictions is shown below:

fully utilized consumption

$$= 241.08 + 0.04 * \textit{Processor Speed} + 9.54 * \textit{Cores} + -2.50 * \textit{Reference Age}$$

Actual vs Predicted consumption are visualized below:

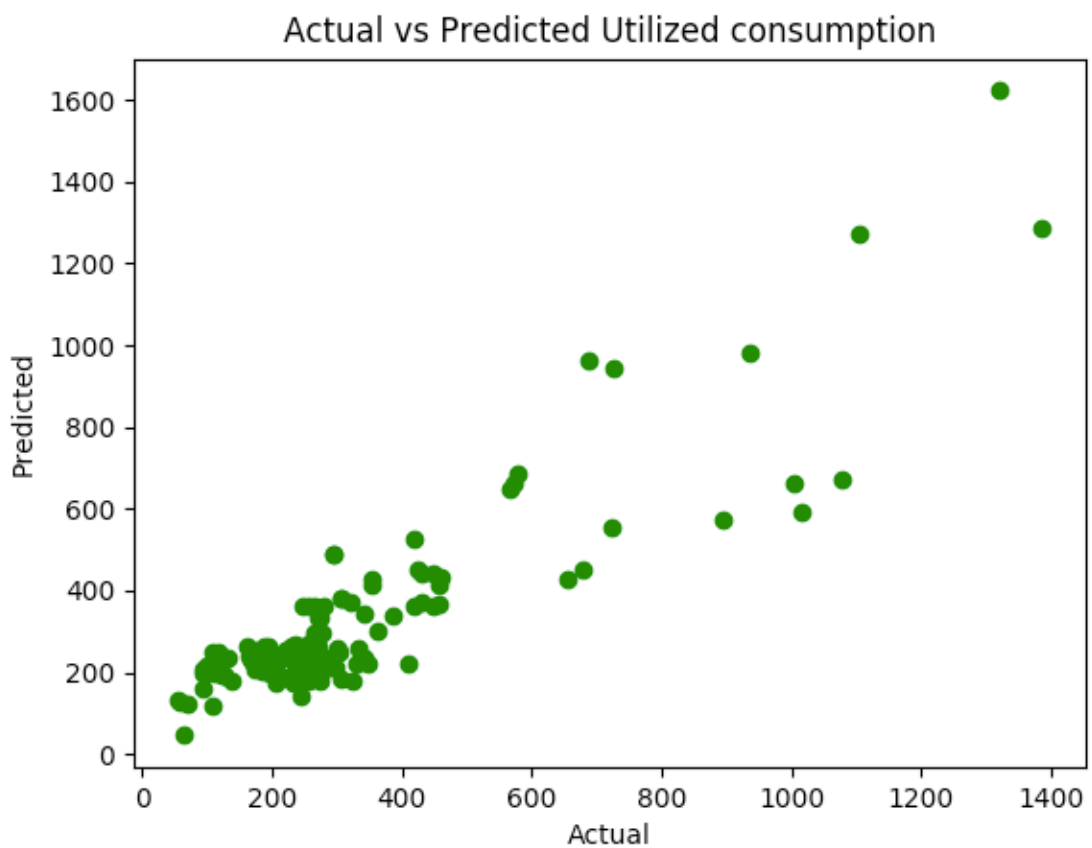


Figure 75 :Actual Vs Predicted

Comparison of Actual Vs Predicted in line chart is presented below:

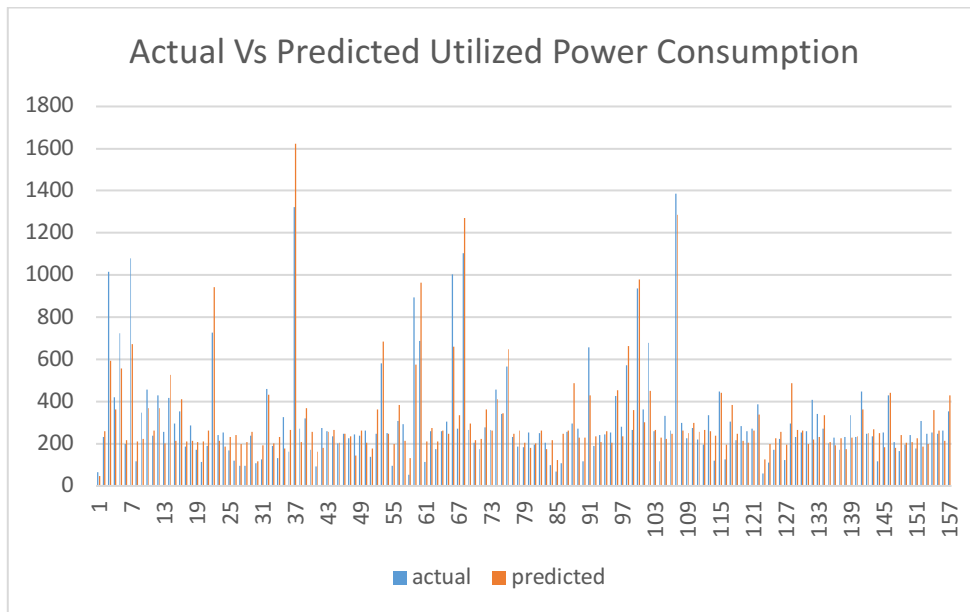


Figure 76:Actual Vs Predicted

Scenario 5

In this scenario, four input attributes were chosen. But they are different to what is used in the scenario 2:

Independent Variable	Dependent Variable
Processor Speed Number of Cores Processor Thread Count Reference Age (months)	Fully Utilized Power Consumption

And the results re shown below:

Independent Variable	Coefficient	Intercept (Constant)	Average Error %
Processor Speed	3.24	210.10	29.54%
Cores	9.70		
Processor Threads	7.14		
Reference Age	-3.12		

Machine Learning Model

Based on the trained dataset, the model used for predictions is shown below:

Average watts when fully utilized

$$\begin{aligned} &= 210.10 + 3.24 * \textit{Processor Speed} + 9.70 * \textit{Cores} \\ &+ 7.14 * \textit{Processor Threads} - 3.12 \\ &* \textit{Reference Age} \end{aligned}$$

Actual vs predicted consumption are visualized below:

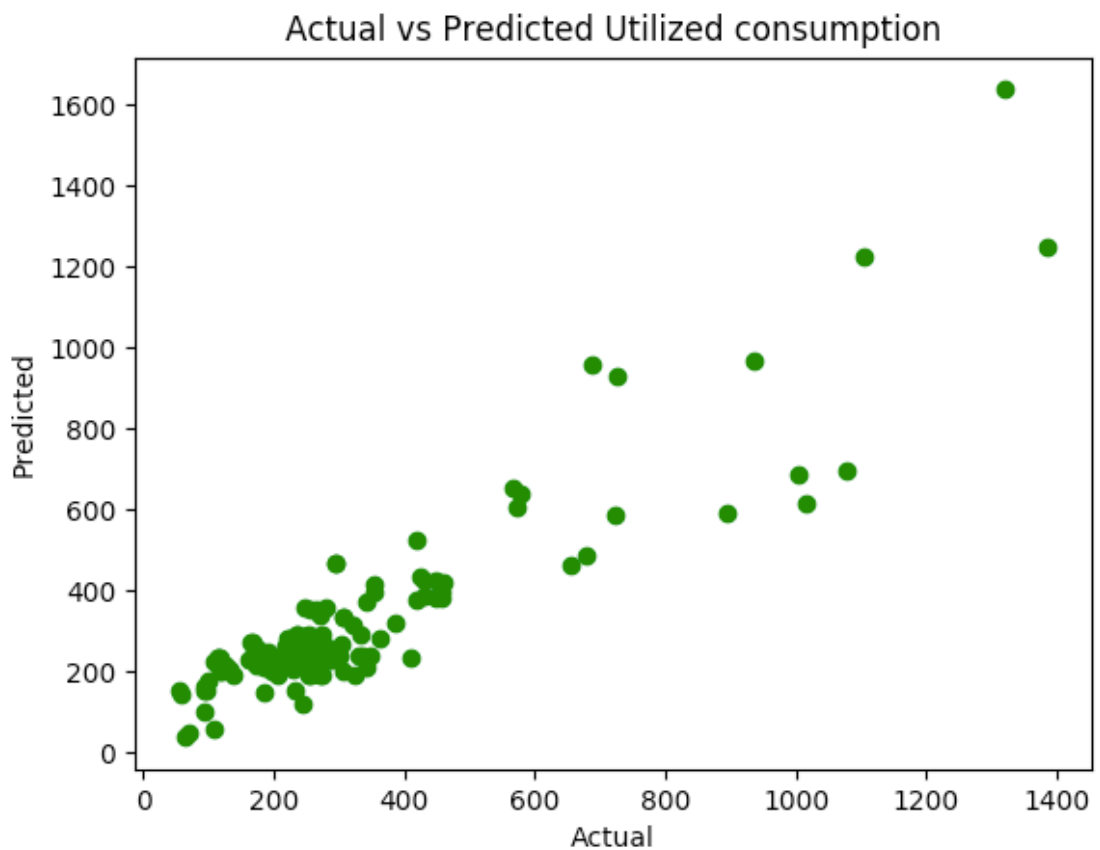


Figure 77: Actual Vs Predicted

Actual Vs Predicted is presented in a line chart below:

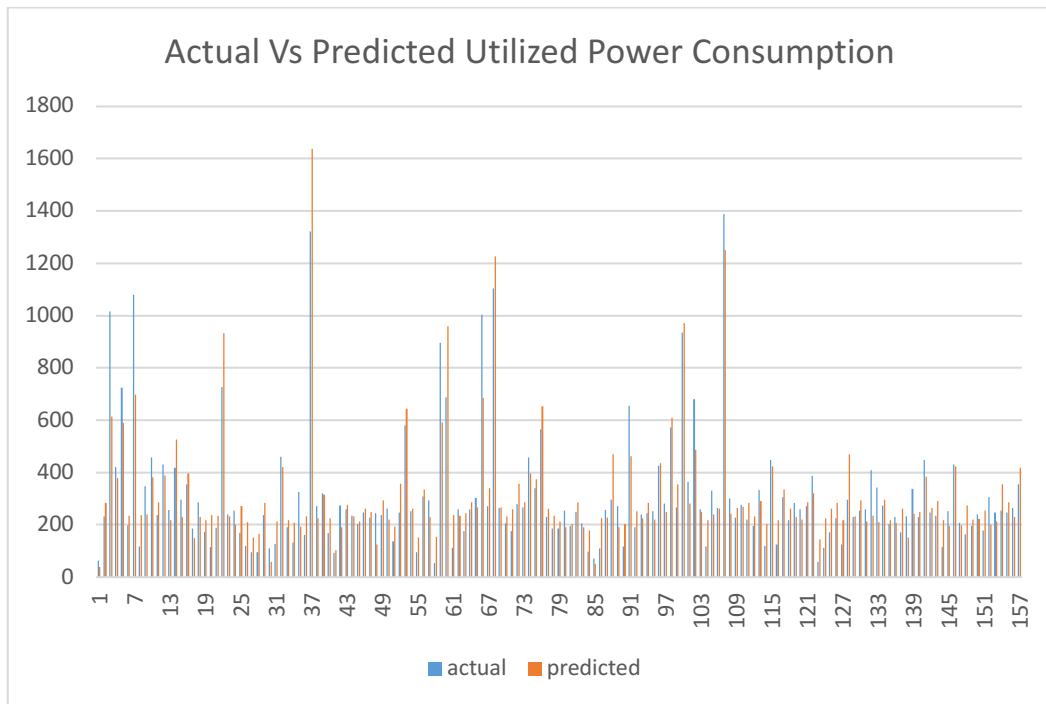


Figure 78 :Actual Vs Predicted

Scenario 6

In the last scenario, all six independent attributes were used in the prediction:

Independent Variable	Dependent Variable
Processor Speed Number of Cores Memory Processor Thread Count Power Rating Reference Age (months)	Fully Utilized Power

The results are shown below:

Independent Variable	Coefficient	Intercept (Constant)	Average Error %
Processor Speed	0.27		

Cores	9.72	254.57	26.80 %
Memory	-1.39		
Processor Threads	5.81		
Power Rating	-2.83		
Reference Age	-0.52		

Machine Learning Model

Based on the trained dataset, the model used for predictions is shown below:

<p>Average watts when fully utilized</p> $= 254.57 + 0.27 * \text{Processor Speed} + 9.72 * \text{Cores}$ $+ -1.39 * \text{Memory} + 5.81 * \text{Processor Threads}$ $+ -2.83 * \text{Power Rating} + -0.52 * \text{Reference Age}$

A visualized graph between actual and prediction power consumption is displayed below:

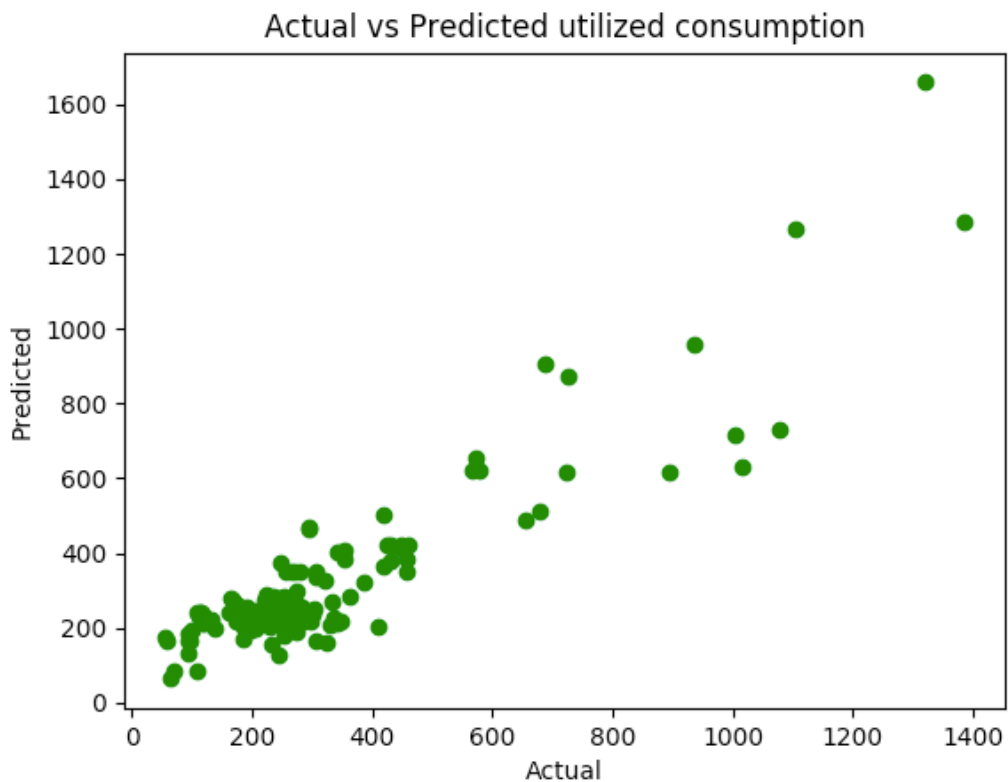


Figure 79: Actual Vs Predicted

Actual Vs Predicted in a line chart below:

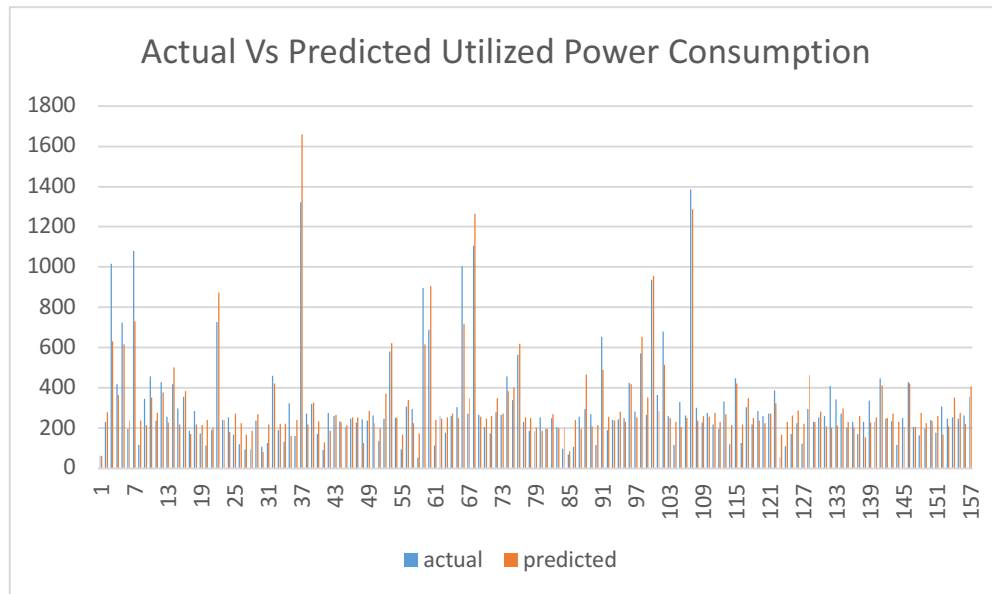


Figure 80:Actual Vs Predicted

8.2.3 Evaluation of Idle Power Consumption

The same as in the prediction of fully utilized power consumption, we have picked a few scenarios for detailed analysis as discussed below:

Scenario 1

We started with number of cores and memory.

Independent Variable	Dependent Variable
Processor Speed	Idle Power Consumption
Number of Cores	

The results are in the table below:

Independent Variable	Coefficient	Intercept (Constant)	Average Error %
Processor Speed	0.01	24.13	55.63%
Number of Cores	0.69		

Machine Learning Model

$$\text{Average power when idle} = 24.13 + 0.01 * \text{Processor Speed} + 0.69 * \text{Cores}$$

The actual and predicted values are visualized in the below graph:

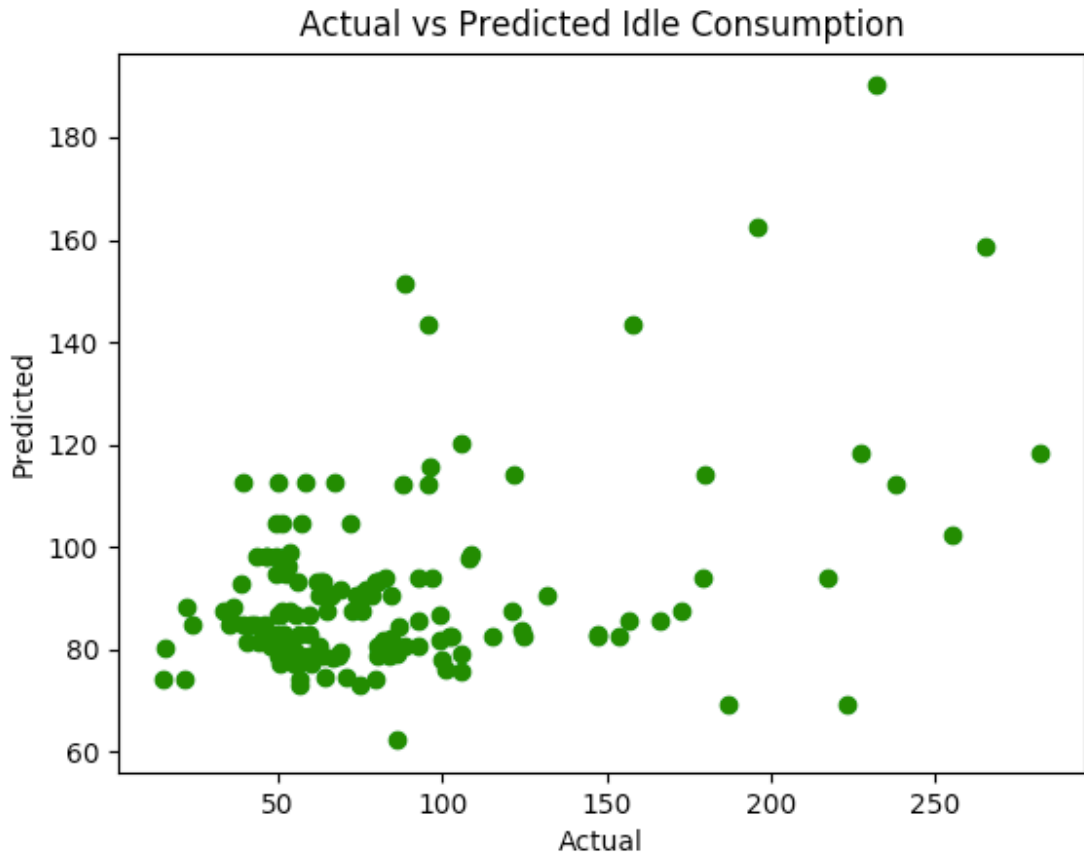
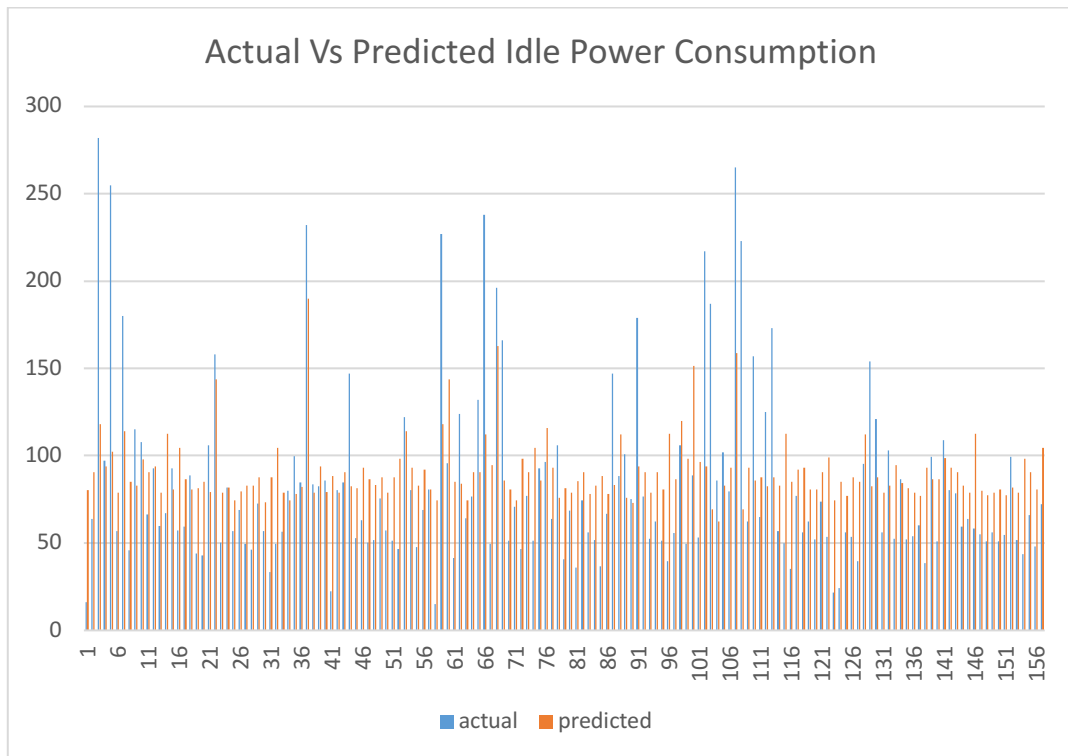


Figure 81: Actual Vs Predicted

Actual Vs Predicted results are visualised in line chart:



Scenario 2

We started with number of cores and memory.

Independent Variable	Dependent Variable
Processor Speed	Idle Power Consumption
Number of Cores	

The results are in the table below:

Independent Variable	Coefficient	Intercept (Constant)	Average Error %
Number of Cores	1.36	75.76	54.22%
Memory	-0.37		

Machine Learning Model

Based on the trained dataset, the model used for predictions is shown below:

$$\text{Average power when idle} = 75.76 + 1.36 * \text{Cores} + -0.37 * \text{Memory}$$

The actual and predicted values are visualized in the below graph:

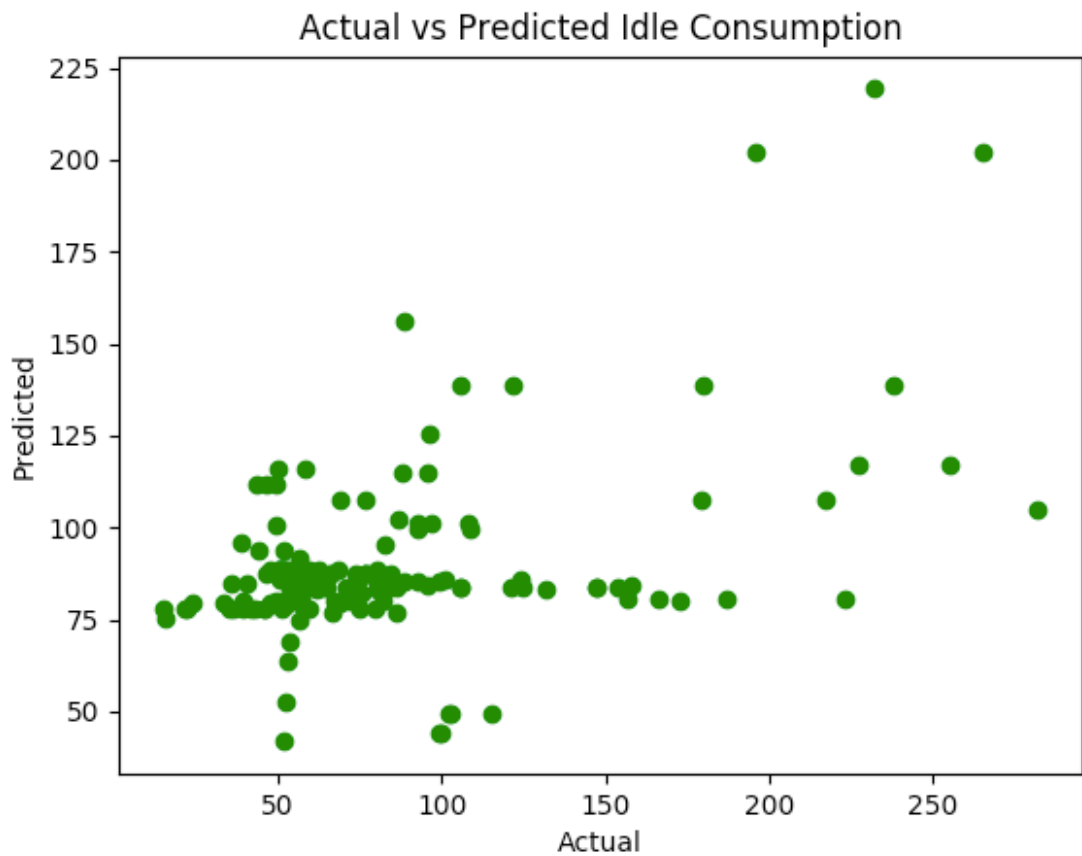


Figure 82:Actual Vs Predicted

Actual Vs Predicted is visualised in a line chart below:

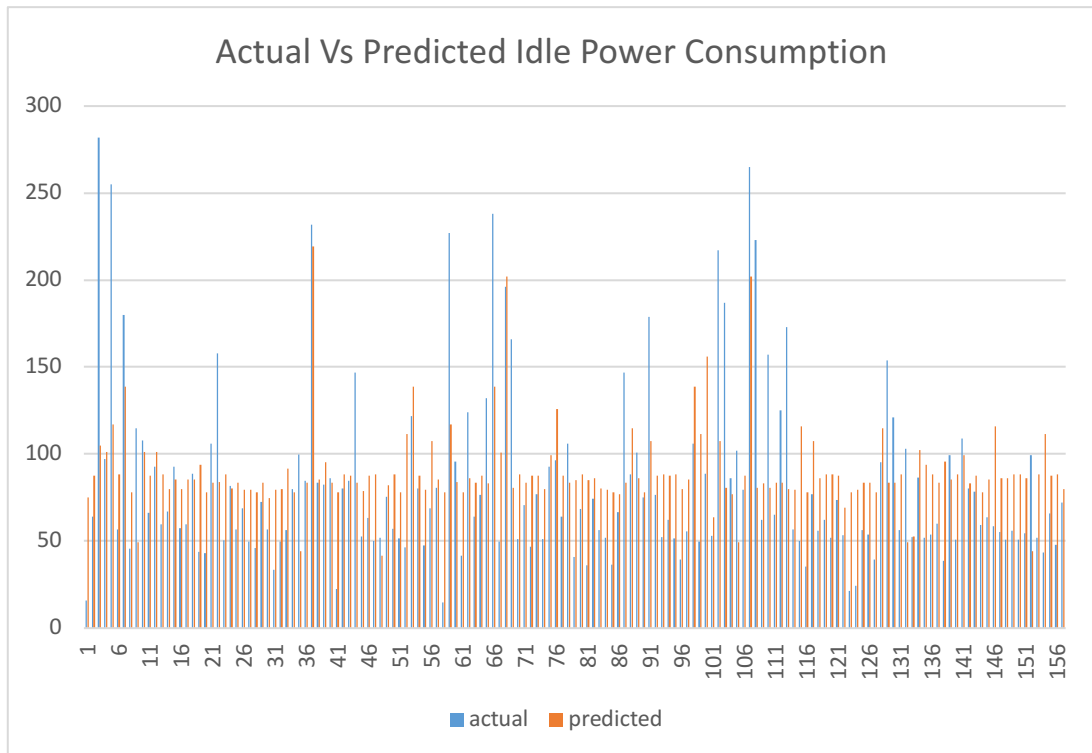


Figure 83:Actual Vs Predicted

Scenario 3

Four input parameters in the first scenario.

Independent Variable	Dependent Variable
Processor Speed	Idle Power Consumption
Number of Cores	
Memory	
Processor Thread Count	
Reference Age (months)	

The results are in the table below:

Independent Variable	Coefficient	Intercept (Constant)	Average Error %
Processor Speed	0.14	267.08	41.05%
Number of Cores	1.61		
Memory	7.90		
Processor Threads	-0.22		
Reference Age	-1.33		

Machine Learning Model

Based on the trained dataset, the model used for predictions is shown below:

<p>Average power when idle =</p> $267.08 + 0.14 * \textit{Processor Speed} + 1.61 * \textit{Cores} + 7.90 * \textit{memory} + -0.22 * \textit{Processor Threads} + -1.33 * \textit{Reference Age}$

The actual and predicted values are visualized in the below graph:

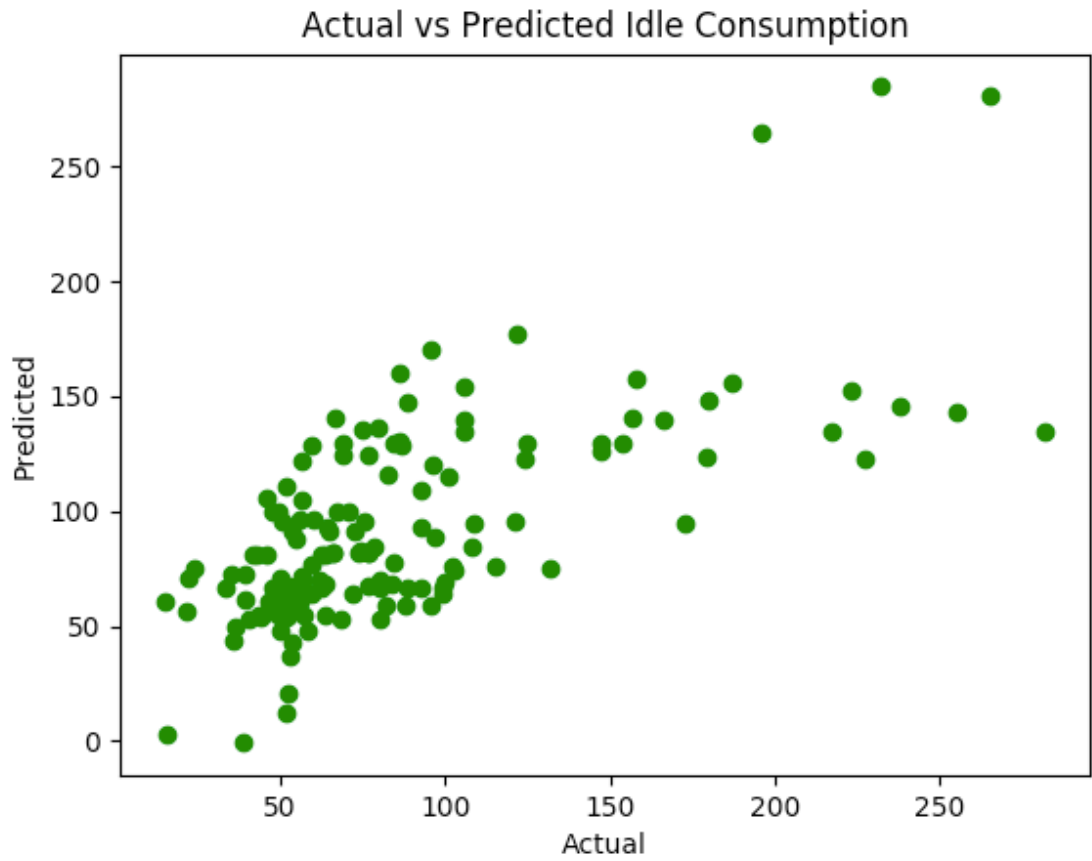


Figure 84: Actual Vs Predicted

Actual Vs Predicted is visualised in a line chart below:

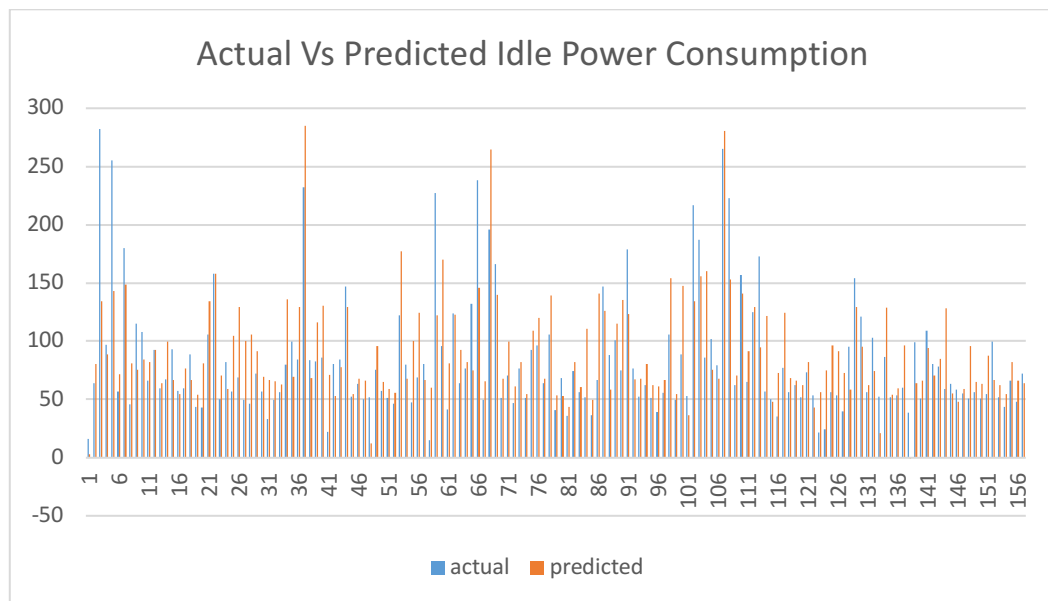


Figure 85: Actual Vs Predicted

Scenario 4

Here all six input attributes were used to feed the algorithm.

Independent variable	Dependent Variable
Processor Speed Number of Cores Memory Processor Thread count Power Rating Reference Age (months)	Idle Power Consumption

The results are in the table below:

Independent Variable	Coefficient	Intercept (Constant)	Average Error %
Processor Speed	- 0.13	263.60	40.78
Number of Cores	1.63		
Memory	7.41		
Processor Thread count	-2.15		
Power Rating	-1.34		
Reference Age	3.47		

Machine Learning Model

Based on the trained dataset, the model used for predictions is shown below:

$$\text{Idle power} = 263.60 + -0.13 * \text{Processor Speed} + 1.63 * \text{Cores} + 7.41 * \text{Memory} + -2.15 * \text{Processor Threads} + -1.34 * \text{Power Rating} + 3.47 * \text{Reference Age}$$

Again, actual vs predicted power consumption is visualized below:

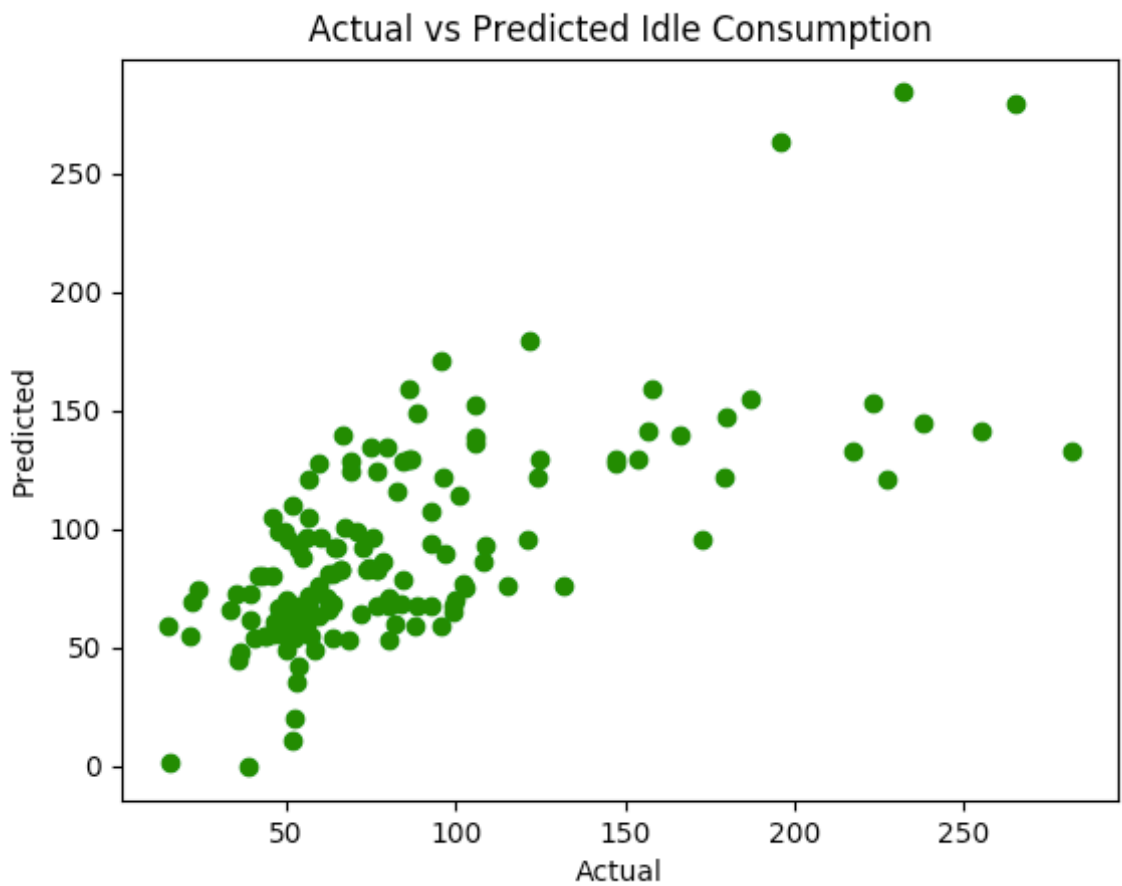


Figure 86:Actual vs Predicted

Actual Vs Predicted in a line chart is presented below:

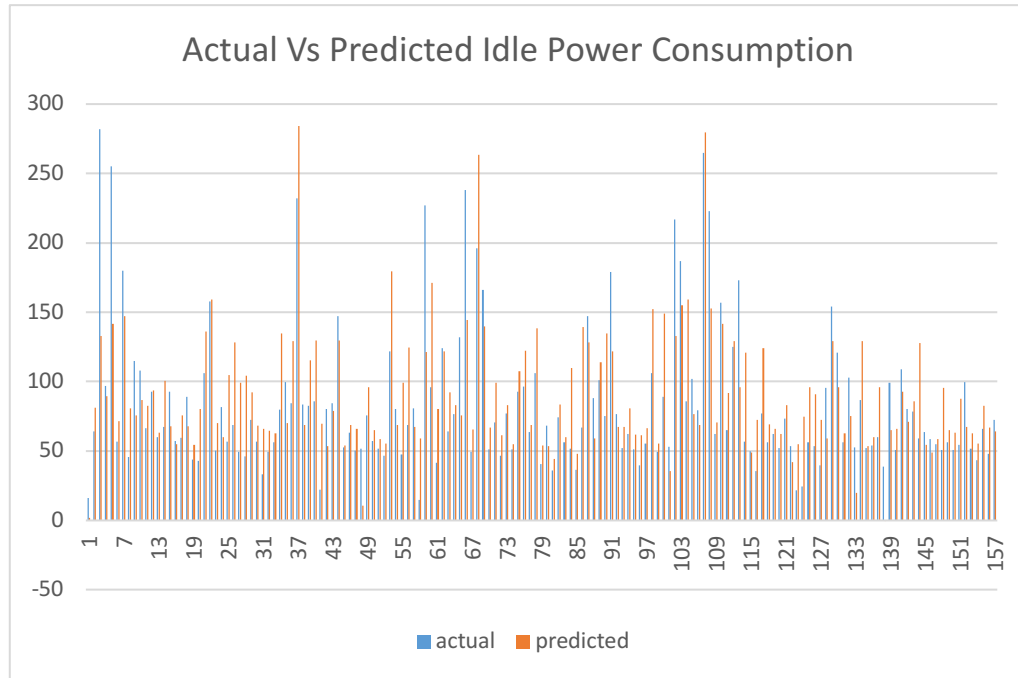


Figure 87: Actual Vs Predicted

8.2.4 Result Comparison of Regression Evaluation

Table 16 presents results of all model evaluations to predict power consumption and the average error percentage:

√ : Included in the Evaluation

× : Not Included in the Evaluation

Processor Speed	Cores	Memory	Processor Threads	Reference Age	Power Rating	Utilized Power Average Error (%)	Idle Power Average Error (%)
Single Independent Parameter							
×	√	×	×	×	×	29.97	55.48
×	×	√	×	×	×	43.29	59.01
×	×	×	×	√	×	54.93	59.56
√	×	×	×	×	×	58.01	61.67
×	×	×	√	×	×	59.29	54.62
×	×	×	×	×	√	69.54	62.27
Combination of Independent Parameters							
√	√	√	√	√	×	29.66	41.05
√	√	√	√	×	×	32.60	47.96
√	√	√	×	×	×	31.82	54.55
√	√	×	×	×	×	31.36	55.63
×	√	√	×	×	×	29.76	54.22
All Independent Parameters							
√	√	√	√	√	√	26.80	40.78

Table 16: Prediction Results Comparison

8.2.5 Conclusion on Regression Result Analysis

Based on the evaluation results, we can draw the below conclusions:

1. The number of cores has the biggest impact on idle power consumption.
2. If you only have one parameter value available, the best candidate to be used as an input attribute is the number of cores. This is the case for both idle and fully utilized consumption.
3. If you have got two variables available, the best prediction results are produced for the combination of number of cores and the memory.
4. The best result which is 26.80% is a decent prediction accuracy. However, this is still too high to use in a production environment.

8.3 Deep Learning Model Evaluation

This section is about how we evaluated the deep learning predictor. The sequential neural network (NN) was trained using the same training data set as used in the regression evaluation.

We used a scenario-based evaluation strategy by changing the combination of the input features. Again, the predicted features are the fully utilized and idle power consumption of the servers. Test results are analysed and discussed under each scenario.

8.3.1 Performance Tuning

The deep learning models were first trained using the training data. The neural network was configured to run multiple times. The improvements were monitored by changing the number of cycles at each run. As a result of that, we found out that the best value for the number of cycles to be 500. We did not see any improvement in the results after that point.

The technical word for a training cycle is an Epoch. Below is an extract of the output of the last execution cycle. We can see in the logs that the value loss which is a major benchmark in results analysis did not improve beyond that point:

Epoch 00499: val_loss did not improve from 0.49853

Epoch 500/500

32/457 [=>.....] - ETA: 0s - loss: 20.0671 - mean_absolute_error: 20.0671

457/457 [=====] - 0s 132us/step - loss: 8.7768 - mean_absolute_error: 8.7768 - val_loss: 7.9127 - val_mean_absolute_error: 7.9127

Epoch 00500: val_loss did not improve from 0.49853

To find the best model configurations, we carried out number of testing scenarios and iterations. Under each scenario, we changed the number of epochs and batch size and then compare the results to achieve best optimized results.

The best performance in other words, the lowest average error percentage was achieved for the below combination:

Epochs – 500

Batch size – 32

An Epoch is a cycle of the entire training data set which is passed forward and backward through the neural network.

The benchmark we used to compare each scenario was the model accuracy of the neural network which is related to the accuracy of the prediction.

For the comparison, we picked two scenarios to discuss details. In both cases, all six independent parameters were used to predict fully utilized power consumption.

Details as below:

Scenario 1

Batch Size	Number of Epoch	Average Error %
16	500	18.41%

This has the least model accuracy and accordingly reported the highest error percentage.

Model accuracy graph displaying the accuracy for each epoch is shown below:

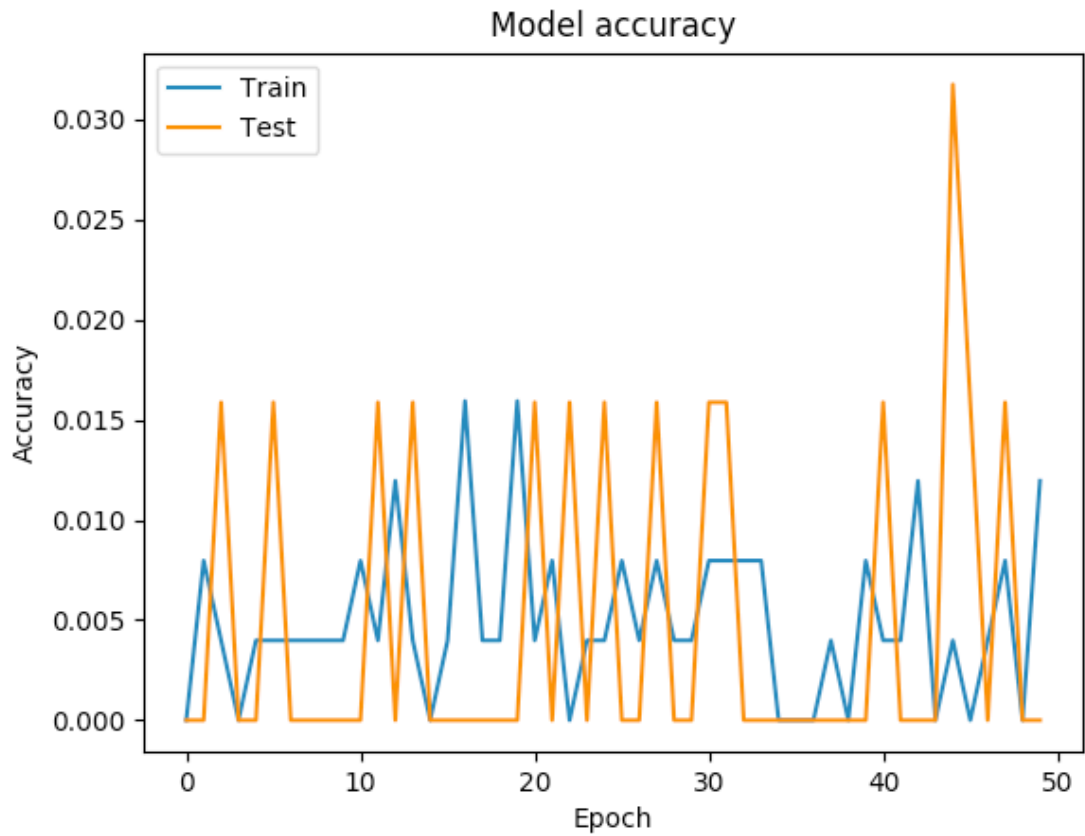


Figure 88: Model Accuracy

Scenario 2

Batch Size	Number of Epoch	Average Error %
32	50	17.87%

When the batch size was increased to 32 without changing the number of epochs, we saw a slight improvement in the accuracy.

The model accuracy graph is shown below:

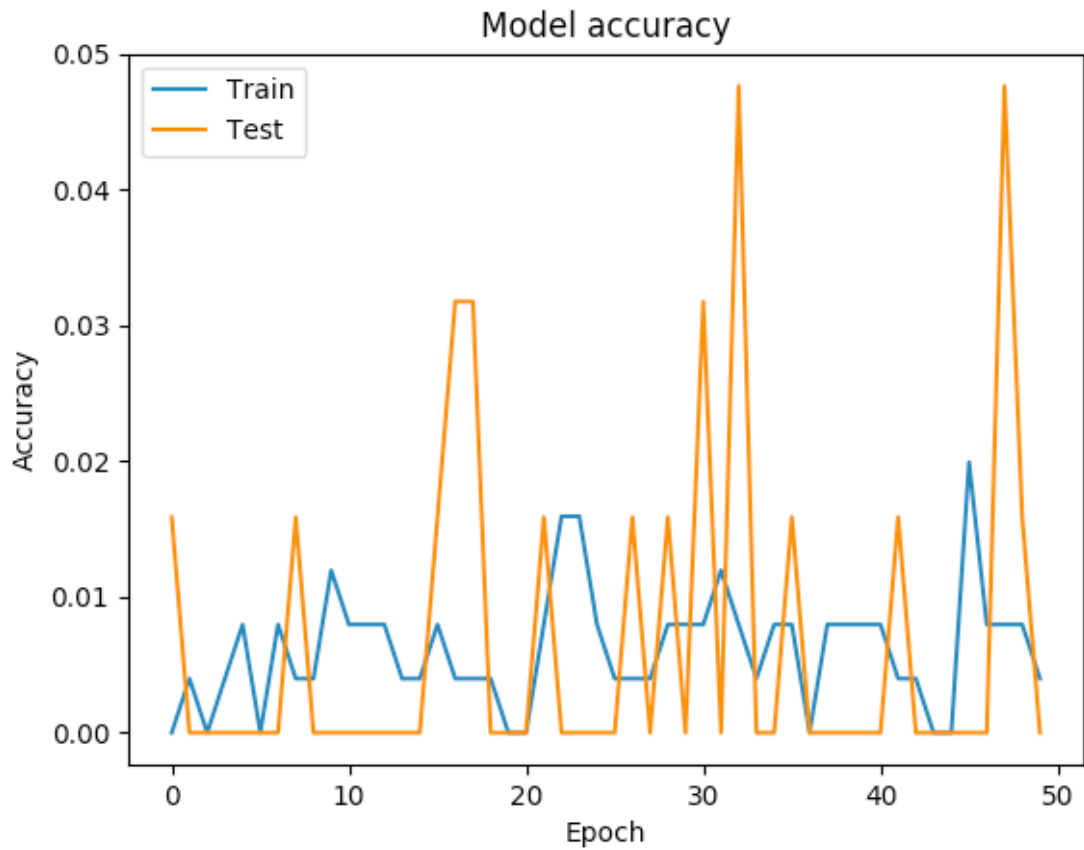


Figure 89: Model Accuracy

Scenario 3

Batch Size	Number of Epoch	Average Percentage	Error
32	500	12.63	

This was the best-case scenario. We saw a significant improvement in the accuracy when number of epochs was increased to 500 which is the optimized value.

Model accuracy graph is shown below:

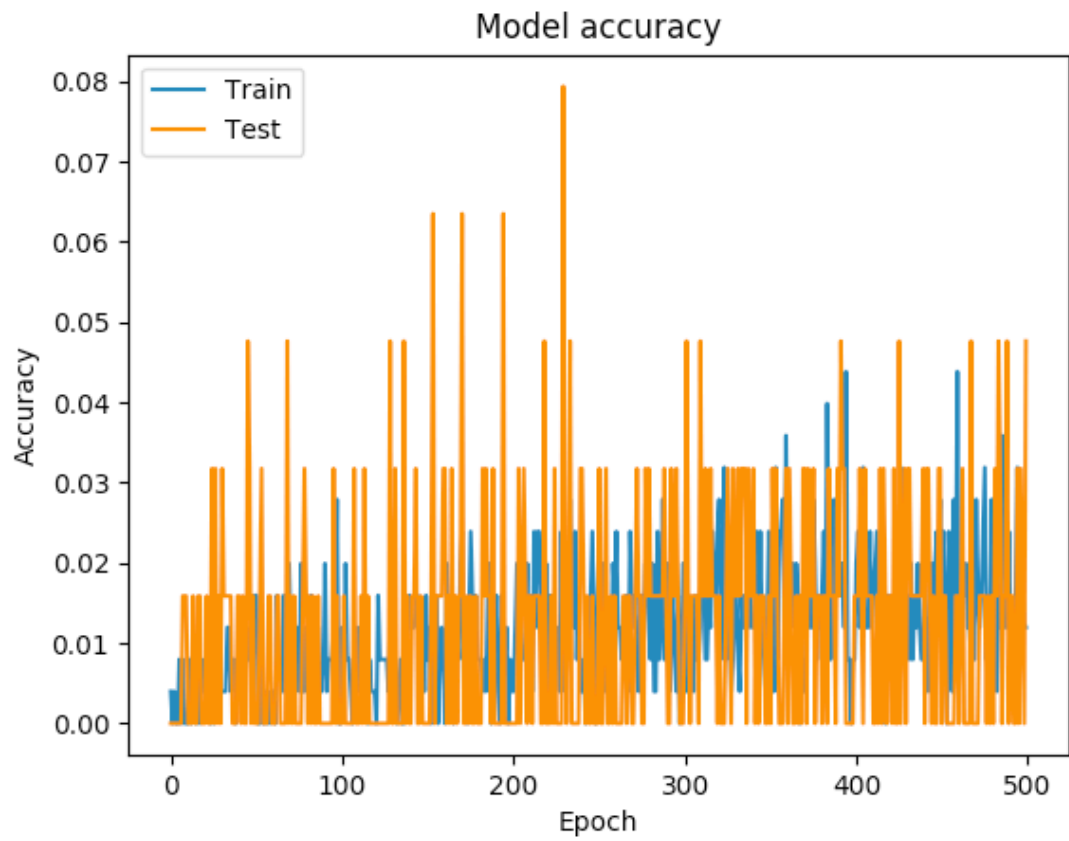


Figure 90: Model Accuracy

8.3.2 Evaluation of Fully Utilized Power Prediction

We picked a few scenarios to study the results of deep learning predictions.

Scenario 1

This is the best-case scenario where we take all six independent parameters for predictions. The summary of the results is shown in the table below:

Independent Variable	Dependent Variable	Average Error %
Processor Speed	fully utilized power consumption	12.63%
Number of cores		
Memory		
Threads per Core		
Reference Age		
Power Supply Rating		

The bellow table shows only the first 10 results:

Actual	Predicted	predict-actual	Error Percentage
173	175.29	2.29	1.32
265	266.15	1.15	0.43
259	260.10	1.10	0.42
307	308.23	1.23	0.40
511	512.55	1.55	0.30
713	716.19	3.19	0.44
559	560.89	1.89	0.33
327	327.91	0.91	0.27
255	254.79	0.20	0.07

The figure below presents the actual values vs predicted energy consumption:

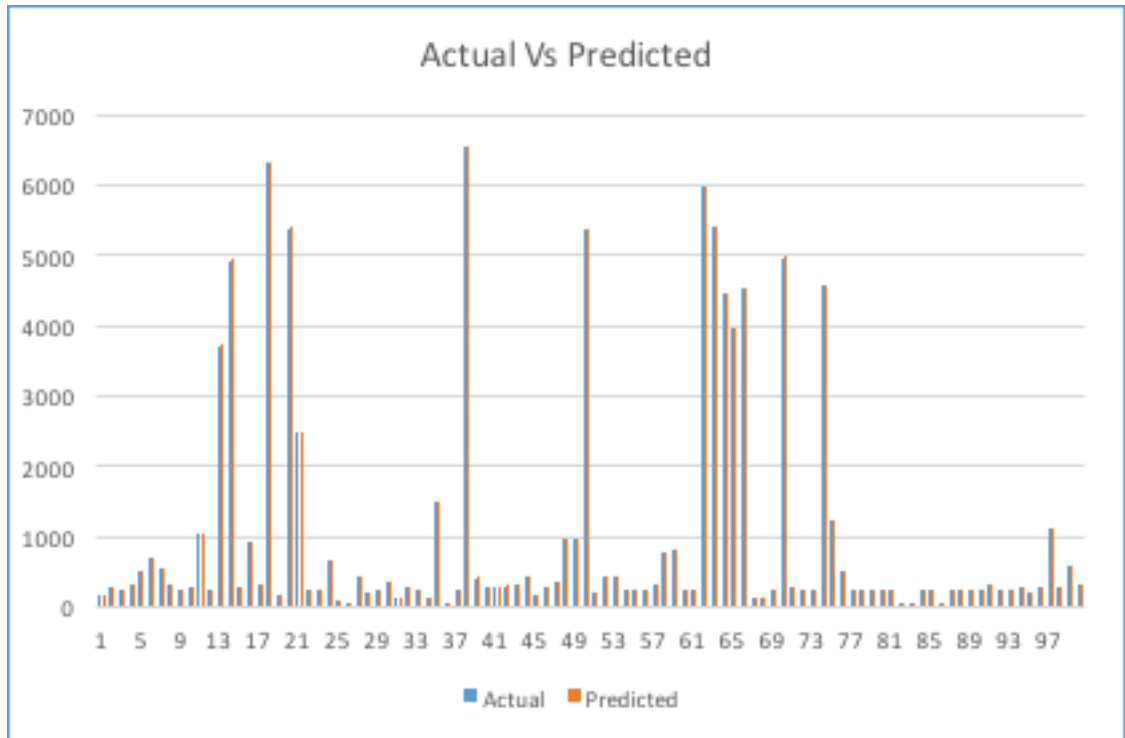


Figure 91 : Actual vs Predicted – Scenario 1

The actual utilized power consumption against the error percentage is presented in figure 92:

Actual Vs Error Percentage

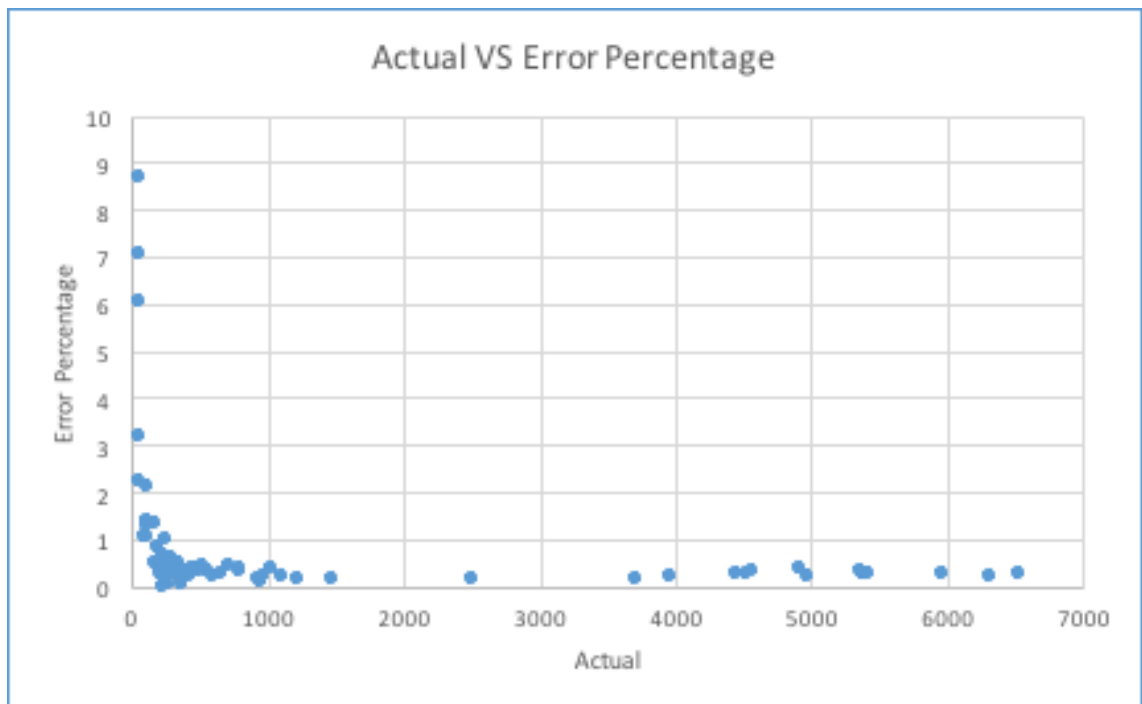


Figure 92: Actual vs Error – Scenario 1

How the model accuracy was performed at each training model training cycle is shown below:

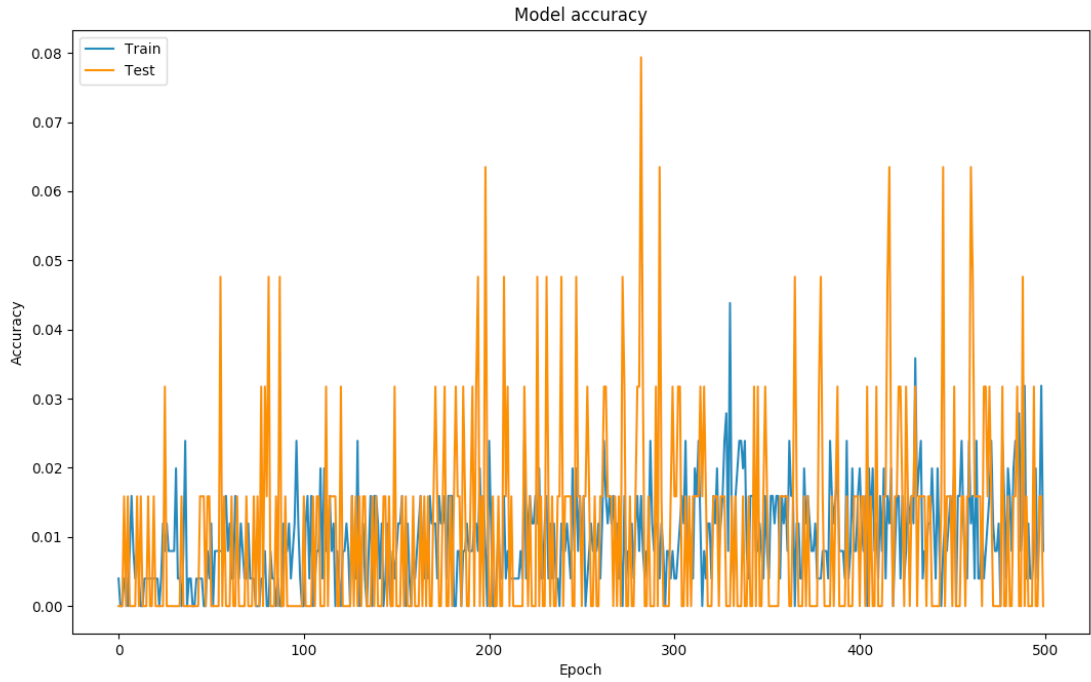


Figure 93:Model Accuracy

Scenario 2

In this scenario, we consider the best results achieving a single independent parameter which is number of cores. The results are in the table below:

Independent Variable	Dependent Variable	Average Error %
Number of Cores	Fully Utilized Power Consumption	20.49%

For each test data point, the actual and the predicted energy consumption are compared in the below graph:

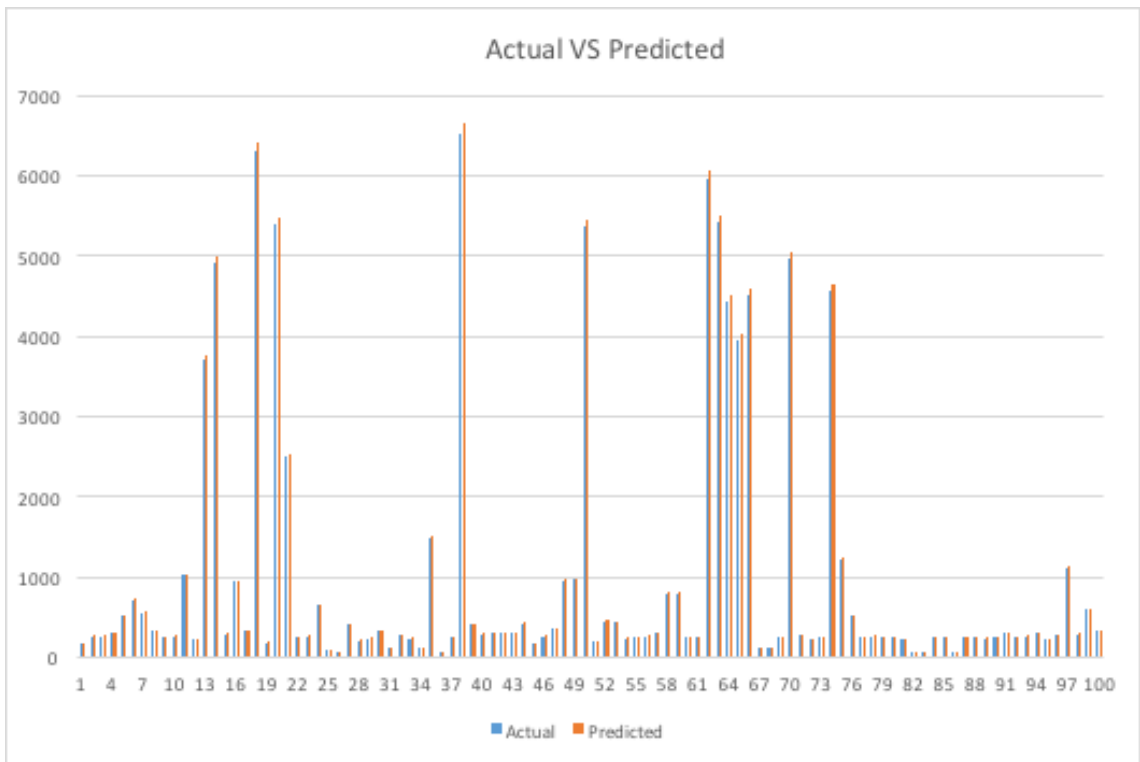


Figure 94:Actual vs Predicted – Scenario 2

The error percentage of the predicted value for each actual power consumption is plotted below:

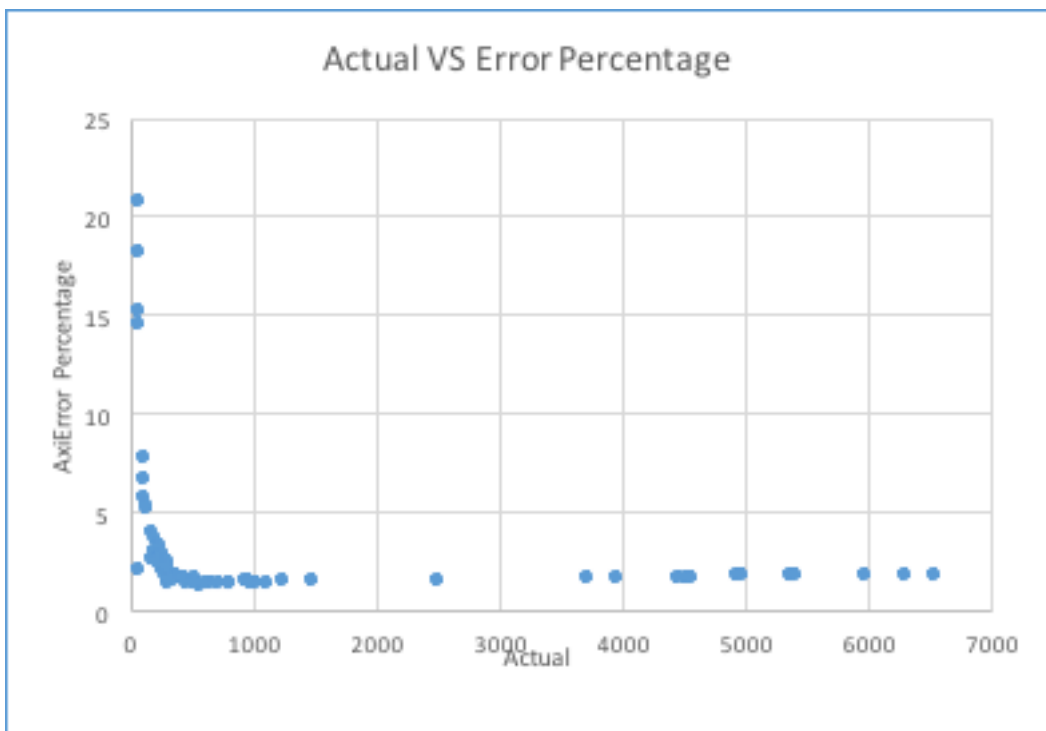


Figure 95:Actual vs Error – Scenario 2

Model Accuracy

The below graph shows the accuracy of the Neural Network at each Epoch. The accuracy is out of 1. The accuracy of 0 means an utterly unsuccessful prediction while 1 is entirely successful.

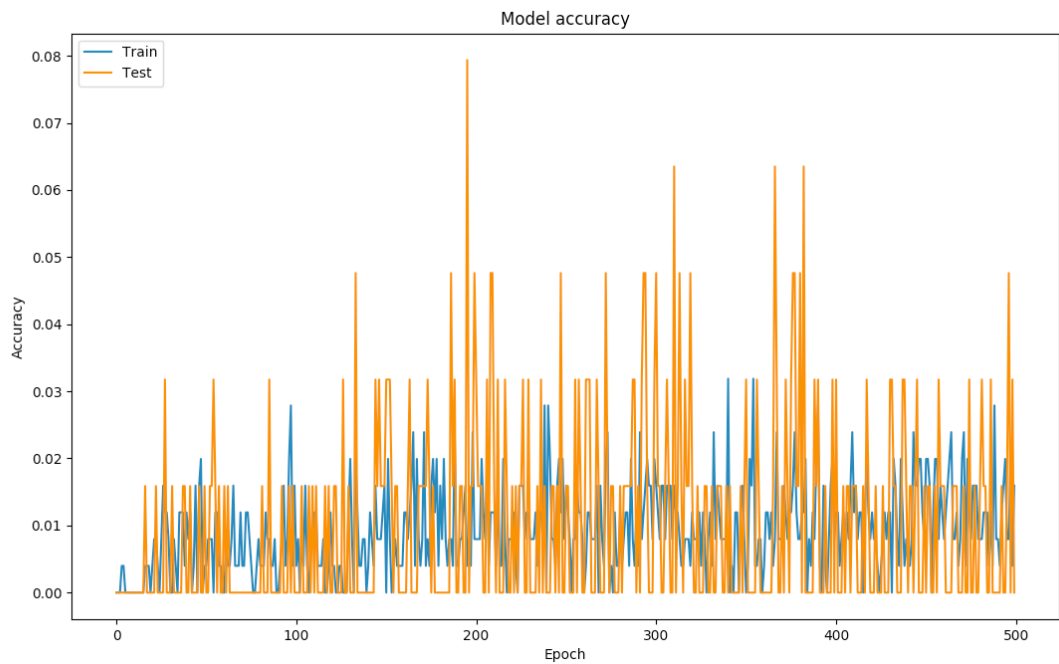


Figure 96 : Model Accuracy

Scenario 3

Here we consider the best results achieving two input parameters We took processor speed and memory as input parameters and executed the models. The results are in the table below:

Independent Variable	Dependent Variable	Average Error %
Processor Speed Memory	Fully Utilized Power Consumption	33.78 %

Actual utilized power against the prediction error percentage is shown in the graph below:

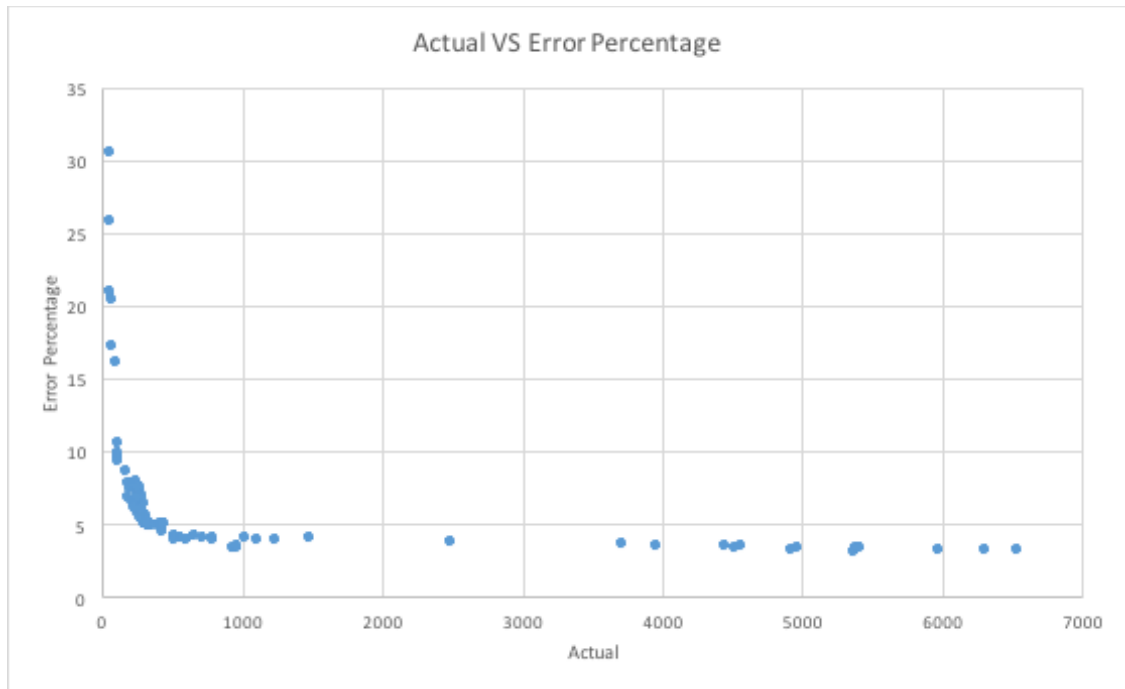


Figure 97: Actual Vs Error Percentage - Scenario 3

The actual utilized power consumption vs predicted values are visualized in the below graph:

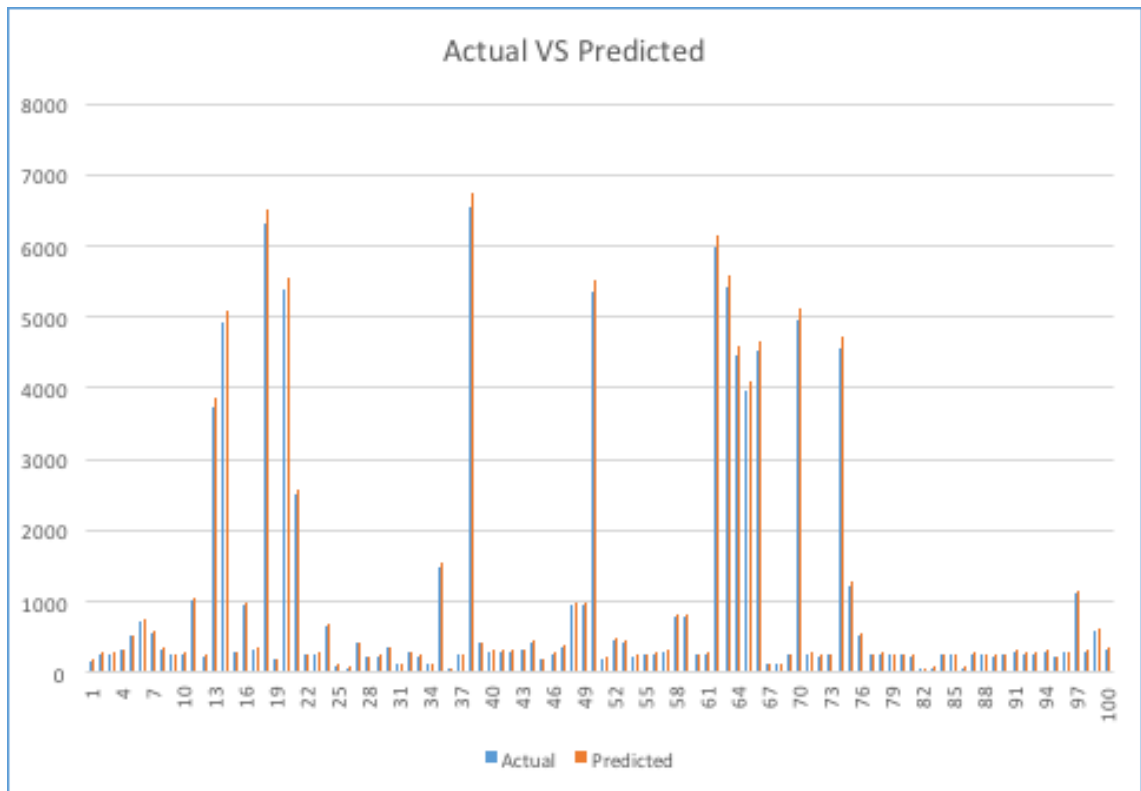


Figure 98: Actual Vs Predicted - Scenario 3

Model Accuracy vs Epoch

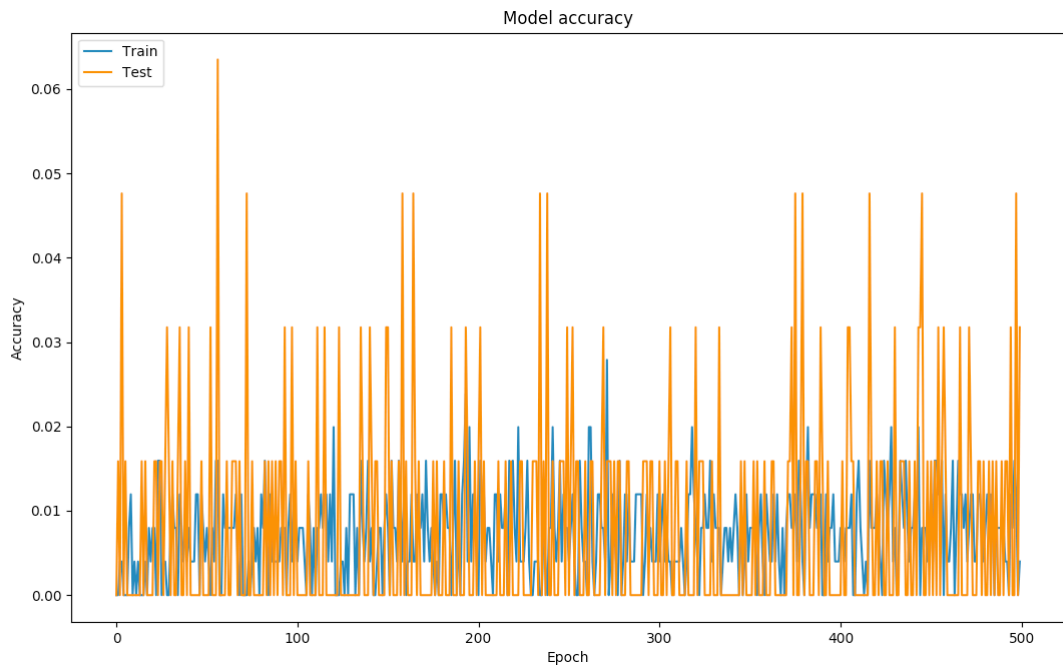


Figure 99: Model Accuracy Vs Epoch – Scenario 3

Scenario 4

Server memory and the reference age were used as independent parameters. The results of the model execution are shown in the table below:

Independent Variable	Dependent Variable	Average Error %
Memory	Fully Utilized Power	26.81%
Reference Age	Consumption	

Table 17: Scenario 1 Results

The actual vs predicted utilized power consumption is visualized in the graph below:

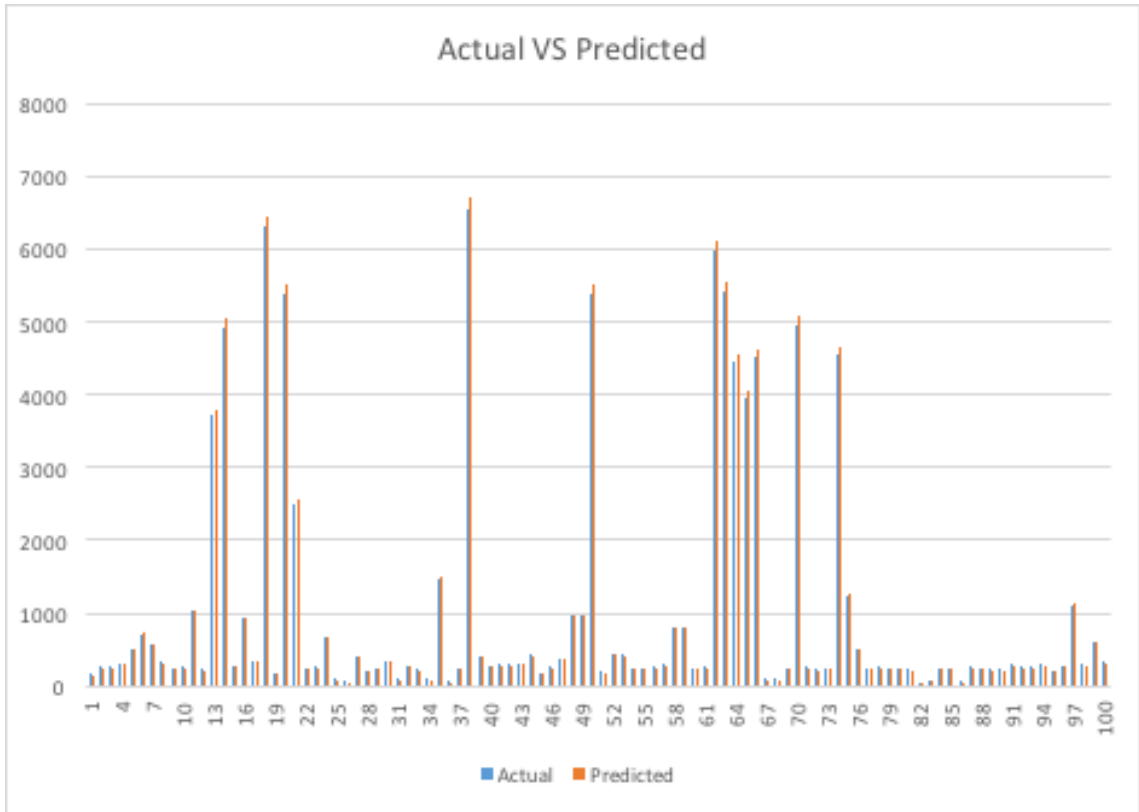


Figure 100:Actual Vs Predicted - Scenario 4

Actual utilized power consumption against predicted error percentage is visualized in the below graph:

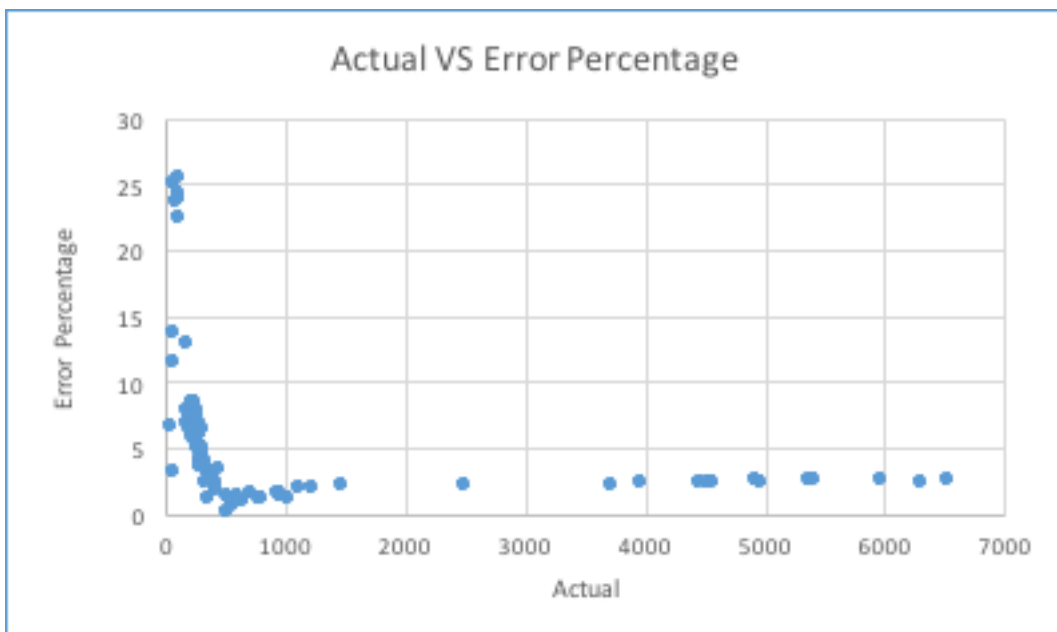


Figure 101 : Actual Vs Error Percentage - Scenario 4

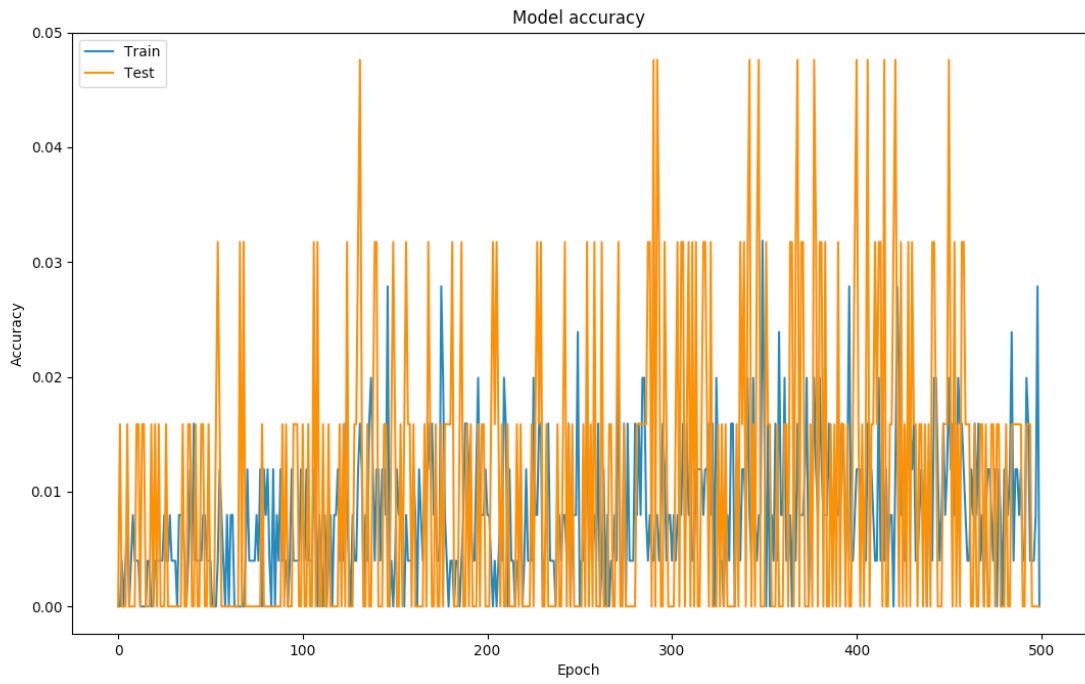


Figure 102:Model Accuracy - Scenario 4

8.3.3 Evaluation of Idle Energy Consumption

We picked some scenarios to study the results of idle energy consumption predictions using deep learning. Details are discussed below:

Scenario 1

Independent variable	Dependent Variable	Average Error %
Processor Speed Number of Cores Memory Reference Age	Fully Utilized Power Consumption	25.35%

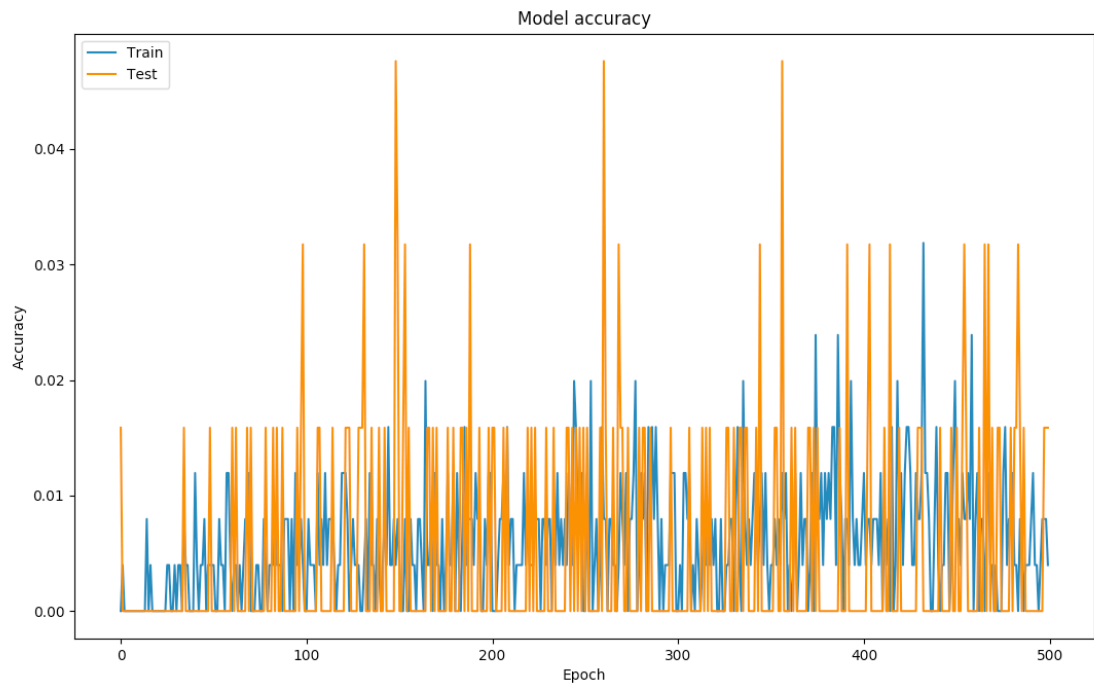
Table 18 : Scenario 1 Results

The First 10 Predictions

Actual	Predicted	Absolute Difference (predict-actual)	Error Percentage
89.4	89.33	0.06	0.07
76.4	76.32	0.07	0.09
76.6	76.52	0.08	0.10
99.5	98.84	0.65	0.65
145	144.95	0.04	0.03
176	175.66	0.34	0.19
127	127.18	0.18	0.14
98.2	97.54	0.65	0.67
68.5	68.01	0.49	0.733

Table 19:First 10 Predictions

Model accuracy across each training cycle is shown below:



8.3.4 Results Comparison - Power Consumption

Table 20 summarises the valuation results for fully utilized power consumption:

√: Included in the Evaluation

×: Not Included in the Evaluation

Processor Speed	Cores	Memory	Threads Per core	Reference Age	Power Supply Rating	Utilized power Average Error (%)	Idle power average error %
All Independent Parameters							
√	√	√	√	√	√	12.63	17.09
Single Independent Parameters							
√	×	×	×	×	×	50.95	41.15
×	√	×	×	×	×	20.49	34.27
×	×	√	×	×	×	28.69	43.42
×	×	×	√	×	×	45.70	46.94

×	×	×	×	√	×	44.46	41.41
×	×	×	×	×	√	24.34	29.48
Combination Independent Parameters							
√	√	√	√	√	×	13.58	21.87
×	√	√	√	√	√	13.22	16.77
√	√	×	×	×	×	18.74	38.19
√	×	√	×	√	√	15.77	22.60
×	√	√	×	×	×	20.26	30.85
√	√	×	×	×	√	16.59	16.59
×	√	√	√	×	×	21.77	39.22
×	√	√	×	×	×	19.68	31.10
×	×	√	√	√	×	25.45	22.88
×	√	√	√	√	×	15.78	21.66

Table 20: Results Comparison: Fully Utilized Power

8.3.5 Analysis of Results

Analysing the evaluation results, we can draw the below conclusions:

1. The best prediction accuracy of 12.63% is for utilized power consumption, when all six independent parameters are provided.
2. If you have only got one input attribute value in hand, the best candidate is
 - Power supply rating for idle power consumption: prediction percent error of 29.48%.
 - Number of cores for fully utilized power consumption: prediction percent error of 20.49%.

8.4 Summary

Analysing the details of the evaluations is presented in this chapter. We showed a linear regression between selected combined features of servers and the power consumption. We also ruled out the usage of single linear regression (SLR) models as there is no strong linear regression relationship between any single feature and the power consumption. This is discussed in detail in the model analysis chapter.

In this chapter, with test evidence, we concluded that machine learning regression models are capable of predicting the power consumption with a good average percent error of 26.80%. However, this level of accuracy is not good enough to use in a production application.

So, we continued the research aiming to improve the accuracy by choosing another ML technique which is Deep Learning. A comprehensive model analysis was carried out first and then it was implemented in a tool. We managed to improve the accuracy by reducing the average percent error to 12.63% using those DL algorithms.

We present comprehensive details of the DL based predictor on three different aspects in the evaluation process. As detailed below, all aspects were satisfied.

1. Accuracy

We managed to achieve 12.63% of average error percentage for the prediction for a large test data set. This is a decent and acceptable accuracy level for a commercial software application.

2. Consistency

Evaluation results were calculated by running the model number of times. Each time the predictions were in line with standard prediction accuracy.

3. Scalability

The Neural network model we built was a significantly complex one with number of layers. However, we showed the evidence of a very efficient execution cycle in the prediction process. This is due to the advanced software engineering principles upon which the tool is built. As an example, multi-threading programming helped the tool to run completely independent multiple prediction processes concurrently, to speed up the workload.

Part V

Conclusion

“Life is trying things to see if they work.”

-Ray Bradbury

CHAPTER IX

CONCLUSION AND FUTURE WORK

9.1 Summary and Conclusion

This research is based on measuring the energy consumption of data centres by applying estimations and predictions. It resulted in a number of research contributions.

We conducted a survey on exiting publications on energy consumption estimations by querying well known information sources. The results and analysis are presented in chapter 2 with the identified gaps.

We then led the research onto address the gaps. Firstly, we performed analysis on how individual features of each DC equipment category contributes to the total energy consumption. We built models for each category and the details are discussed in the chapter 3.

The best way of evaluating models is by creating a tool to incorporate them. That way, real-world users could get involved to use on a day-to-day basis. So, we built an online tool by utilizing state-of-the art architectural principles and technologies. The tool scales very well based on the demand, meaning it has the capability to serve many users at the same time, without compromising the performance and the user experience.

Back-end service functions of the application were utilized to evaluate those models. A combination of unit testing and integration testing strategies were used in the evaluation strategy. Individual test cases were written using Junit. Mockito framework was used to mock some of the dependencies which business logic does not directly depend on.

Implementation and evaluation details are presented in chapter 6.

Since the servers are the single most energy consumers of the DC IT equipment, the focus of the research was directed to measuring the energy consumption of servers. Even though the models we developed can be used to calculate the total server energy consumption, there is a practical obstacle. Idle and utilized power rates of the servers are some input parameters of the total energy models but are not easy to find out for some servers. This is especially for the custom-built ones which are very common in the DCs.

There public data sources where these figures are published but only for the standard specifications. Spec.org is a reliable public source.

To address this problem, we leveraged the machine AI and in particular learning techniques to predict the energy consumption of such non-standard servers, using the published data for well-known specifications. The rest of this research is based on the analysis, implementation and evaluation of the machine learning models in predicting the idle and fully utilized energy consumption of servers.

Chapter 4 is on the analysis of Linear Regression models and a comparison on simple and multi linear regression models. With the evidence, we showed that the MLR was the better fitting model than SLR for predicting the consumption using training data.

We built a proof of concept (POC) tool for implementation machine learning models and algorithms. It is the first version of MALEP (Machine Learning Energy Predictor) which we have later extended to incorporate deep learning algorithms as well.

After reviewing the evaluation results of the regression based MALEP - version 1, we decided we needed to improve the accuracy if we wanted to make the tool available for real users. With that in mind, research was carried out on deep learning. Details on the analysis of deep learning models which is the MALEP version 2 is discussed in chapter 5.

Chapter 7 is on the MALEP implementation, covering both version 1 and 2. Evaluation of MALEP models are presented in chapter 8.

With the DL models, we managed to predict the server energy consumption with an average error percentage of 12% which is a decent and acceptable accuracy for real world applications.

Due to the successful prediction rate we managed to achieve, we decided to make it open source software. It is available to download from GitHub and ready to use for data centre operators. They can use it to find out idle and utilized power consumption of their servers. The results can be fed into our model-based web tool to find out total energy consumption of the data centre.

9.2 Future Perspectives

This research provides a lead for a lot of future work, both short-term and long-term.

9.2.1 Short Term

We leveraged AI algorithms to predict server energy consumption. As a short-term extension, the same process can be applied to other IT equipment in the data centres like storage and network equipment as well.

9.2.2 Long Term

As a long-term extension, AI algorithms can be used to predict the energy consumption of data centre infrastructure equipment like cooling as well. Cooling is the single most energy consuming aspect, so having the capability of measuring it will certainly help to apply efficiency measures. That, in effect, should have a significant and positive impact on the environment since the power consumption by data centres has been going up dramatically due to ever-increasing demand.

The energy calculation system we built, captures and stores the structured data for each equipment type. Energy consumption profiles which are constructed out of this data are also stored alongside the generic energy profiles. Machine learning and other AI algorithms can then be used to generate recommendations on power saving as long-term future work.

Techniques like neural networks can be employed to train on the collected data, and the generated recommendations can be sent to data centre managers. They can apply them

to optimize their power savings. Feedback from their experience of the changes in the production environment can also be fed back to the system to improve the ML models. Figure 105 depicts the high-level architecture of the proposed recommendation system.

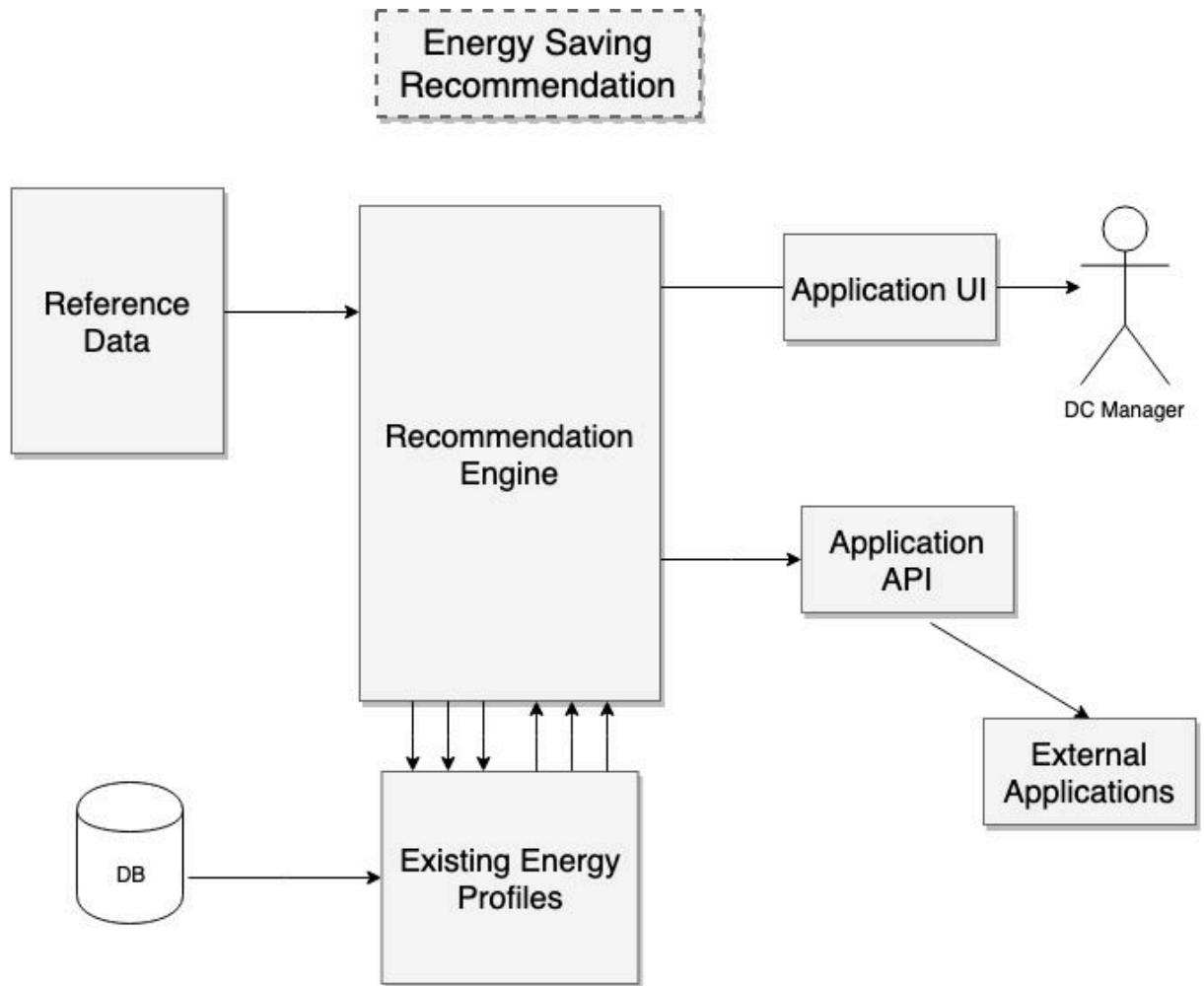


Figure 105:Architecture of proposed Recommendation System

Most important component of the system is the recommendation engine which is fed data from the persistence database. Energy profiles and reference data are loaded into the engine. It generates recommendations on potential energy efficiencies and the users are able to see them using application user interface. The external systems which are integrated are also fed through the application API. The application API can be Restful web service or something similar.

REFERENCES

- G. P. Brady, N. Kapur, J. Summers, and H. Thompson, "A Case Study and Critical Assessment in Calculating Power Utilisation Effectiveness for a Data Centre," 2015.
- S. Rivoire, P. Ranganathan, H.-P. Labs, and C. Kozyrakis, "A Comparison of High-Level Full-System Power Models," p. 5, 2008.
- T. L. Vasques, P. S. Moura, and A. T. de Almeida, "Energy efficiency insight into small and medium data centres: A comparative analysis based on a survey," p. 10.
- M. Kawato, K. Furukawa, and R. Suzuki, "A hierarchical neural-network model for control and learning of voluntary movement," *Biol. Cybern.*, vol. 57, no. 3, pp. 169–185, Oct. 1987.
- A. Hankel, L. Oud, M. Saan, and P. Lago, "A Maturity Model for Green ICT: The case of the SURF Green ICT Maturity Model," p. 8, 2014.
- J. Arjona Aroca, A. Chatzipapas, A. Fernández Anta, and V. Mancuso, "A Measurement-based Analysis of the Energy Consumption of Data Center Servers," in *Proceedings of the 5th International Conference on Future Energy Systems*, New York, NY, USA, 2014, pp. 63–74.
- H. F. Hamann, "A Measurement-Based Method for Improving Data Center Energy Efficiency," in *2008 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (suc 2008)*, 2008, pp. 312–313.
- R. Basmadjian, N. Ali, F. Niedermeier, H. de Meer, and G. Giuliani, "A Methodology to Predict the Power Consumption of Servers in Data Centres," in *Proceedings of the 2Nd International Conference on Energy-Efficient Computing and Networking*, New York, NY, USA, 2011, pp. 1–10.
- M. D. Odom and R. Sharda, "A neural network model for bankruptcy prediction," in *1990 IJCNN International Joint Conference on Neural Networks*, 1990, pp. 163–168 vol.2.
- Y. Xu, J. Du, L. Dai, and C. Lee, "A Regression Approach to Speech Enhancement Based on Deep Neural Networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 1, pp. 7–19, Jan. 2015.
- A. Nouredine, R. Rouvoy, and L. Seinturier, "A review of energy measurement approaches," *SIGOPS Oper. Syst. Rev.*, vol. 47, no. 3, pp. 42–49, Nov. 2013.
- F. Fabris, J. P. de Magalhães, and A. A. Freitas, "A review of supervised machine learning applied to ageing research," *Biogerontology*, vol. 18, no. 2, pp. 171–188, 2017.
- S. M. Abdulhamid *et al.*, "A Review on Mobile SMS Spam Filtering Techniques," *IEEE Access*, vol. 5, pp. 15650–15666, 2017.

G. Kaya Uyanik and N. Güler, “A Study on Multiple Linear Regression Analysis,” *Procedia - Social and Behavioral Sciences*, vol. 106, pp. 234–240, Dec. 2013.

S. Sun, “A survey of multi-view machine learning,” p. 13.

T. T. T. Nguyen and G. Armitage, “A survey of techniques for internet traffic classification using machine learning,” *IEEE Communications Surveys Tutorials*, vol. 10, no. 4, pp. 56–76, Fourth 2008.

A.-C. Orgerie, M. D. de Assuncao, and L. Lefevre, “A survey on techniques for improving the energy efficiency of large-scale distributed systems,” *ACM Comput. Surv.*, vol. 46, no. 4, pp. 1–31, Mar. 2014.

A. J. Smola and B. Schölkopf, “A tutorial on support vector regression,” *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, Aug. 2004.

P. Bemis, “Accurately Predicting the Energy Consumption of Your Data Center -,” *Enterprise Systems*, 2012.: <https://esj.com/articles/2012/12/07/predicting-energy-consumption.aspx>.

“All Published SPEC SPECpower_ssj2008 Results.”
https://www.spec.org/power_ssj2008/results/power_ssj2008.html.

The Eureca Project: <https://dceureca.eu>

V. N. Vapnik, “An overview of statistical learning theory,” *IEEE Trans. Neural Netw.*, vol. 10, no. 5, pp. 988–999, Sep. 1999.

R. Lent, “Analysis of an Energy Proportional Data Center,” *Ad Hoc Netw.*, vol. 25, no. PB, pp. 554–564, Feb. 2015.

B. Whitehead, D. Andrews, A. Shah, and G. Maidment, “Assessing the environmental impact of data centres part 1: Background, energy use and metrics,” *Building and Environment*, vol. 82, pp. 151–159, Dec. 2014.

S. Greenberg, E. Mills, B. Tschudi, and P. Rumsey, “Best Practices for Data Centers: Lessons Learned from Benchmarking 22 Data Centers,” 2006.

A. Dasgupta, Y. V. Sun, I. R. König, J. E. Bailey-Wilson, and J. D. Malley, “Brief review of regression-based and machine learning methods in genetic epidemiology: the Genetic Analysis Workshop 17 experience,” *Genet. Epidemiol.*, vol. 35, no. S1, pp. S5–S11, 2011.

A. Liaw and M. Wiener, “Classification and Regression by RandomForest,” *Forest*, vol. 23, Nov. 2001.

C. Kampichler, R. Wieland, S. Calmé, H. Weissenberger, and S. Arriaga-Weiss, “Classification in conservation biology: A comparison of five machine-learning methods,” *Ecological Informatics*, vol. 5, pp. 441–450, Nov. 2010.

A. Soofi and A. Awan, “Classification Techniques in Machine Learning: Applications and Issues,” *Journal of Basic & Applied Sciences*, vol. 13, pp. 459–465, Aug. 2017.

- C. NIEUWELING, “Code of Conduct for Energy Efficiency in Data Centres,” *EU Science Hub - European Commission*, 14-Nov-2016.: <https://ec.europa.eu/jrc/en/energy-efficiency/code-conduct/datacentres>.
- E. Alpaydm, “Combined 5×2 cv F Test for Comparing Supervised Classification Learning Algorithms,” *Neural Computation*, vol. 11, no. 8, pp. 1885–1892, Nov. 1999.
- W. L. Bircher and L. K. John, “Complete System Power Estimation Using Processor Performance Events,” *IEEE Transactions on Computers*, vol. 61, no. 4, pp. 563–577, Apr. 2012.
- “Complete System Power Estimation: A Trickle-Down Approach Based on Performance Events - IEEE Conference Publication.”: <https://ieeexplore.ieee.org/abstract/document/4211032>.
- K. J. Preacher, P. J. Curran, and D. J. Bauer, “Computational Tools for Probing Interactions in Multiple Linear Regression, Multilevel Modeling, and Latent Curve Analysis,” *Journal of Educational and Behavioral Statistics*, vol. 31, no. 4, pp. 437–448, Dec. 2006.
- K. H. Zou, K. Tuncali, and S. G. Silverman, “Correlation and Simple Linear Regression,” *Radiology*, vol. 227, no. 3, pp. 617–628, Jun. 2003.
- A. Qureshi, R. Weber, H. Balakrishnan, J. Gutttag, and B. Maggs, “Cutting the Electric Bill for Internet-Scale Systems,” p. 12., 2009
- P. Wang, L. Rao, X. Liu, and Y. Qi, “D-Pro: Dynamic Data Center Operations With Demand-Responsive Electricity Prices in Smart Grid,” *IEEE Transactions on Smart Grid*, vol. 3, no. 4, pp. 1743–1754, Dec. 2012.
- M. Dayarathna, Y. Wen, and R. Fan, “Data Center Energy Consumption Modeling: A Survey,” *IEEE Commun. Surv. Tutorials*, vol. 18, no. 1, pp. 732–794, 2016.
- R. Bashroush, E. Woods, and A. Nouredine, “Data Center Energy Demand: What Got Us Here Won’t Get Us There,” *IEEE Software*, vol. 33, pp. 18–21, Mar. 2016.
- K. G. Brill, “Data Center Energy Efficiency and Productivity,” p. 10.
- D. Cole, “Data Center Infrastructure Management,” p. 19.
- T. Daim, J. Justice, M. Krampits, M. Letts, G. Subramanian, and M. Thirumalai, “Data center metrics,” *Management of Environmental Quality: An International Journal*, Sep. 2009.
- L. Newcombe, “Data centre energy efficiency metrics,” p. 54.
- K. H. Tae, Y. Roh, Y. H. Oh, H. Kim, and S. E. Whang, “Data Cleaning for Accurate, Fair, and Robust Models: A Big Data - AI Integration Approach,” *arXiv:1904.10761 [cs]*, Apr. 2019.

- “Deep ensemble learning of sparse regression models for brain disease diagnosis - ScienceDirect.”:
<https://www.sciencedirect.com/science/article/abs/pii/S1361841517300166>.
- Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015.
- J. Schmidhuber, “Deep learning in neural networks: An overview,” *Neural Networks*, vol. 61, pp. 85–117, Jan. 2015.
- L. Deng and D. Yu, “Deep Learning: Methods and Applications,” *SIG*, vol. 7, no. 3–4, pp. 197–387, Jun. 2014.
- Q. Wang, J. Wan, and Y. Yuan, “Deep Metric Learning for Crowdedness Regression,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 10, pp. 2633–2643, Oct. 2018.
- “Design Recommendations for High-Performance Data Centers - Rocky Mountain Institute.”: <https://rmi.org/insight/design-recommendations-for-high-performance-data-centers/>.
- “Dig more coal -- the PCs are coming.”:
<https://www.forbes.com/forbes/1999/0531/6311070a.html#400b659e2580>.
- R. Stanojevic and R. Shorten, “Distributed Dynamic Speed Scaling,” in *2010 Proceedings IEEE INFOCOM*, San Diego, CA, USA, 2010, pp. 1–5.
- T.-Y. Liu, W. Chen, and T. Wang, “Distributed Machine Learning: Foundations, Trends, and Practices,” in *Proceedings of the 26th International Conference on World Wide Web Companion*, Republic and Canton of Geneva, Switzerland, 2017, pp. 913–915.
- B. Heller *et al.*, “ElasticTree: Saving Energy in Data Center Networks,” p. 16.
- D. C. Park, M. A. El-Sharkawi, R. J. Marks, L. E. Atlas, and M. J. Damborg, “Electric load forecasting using an artificial neural network,” *IEEE Transactions on Power Systems*, vol. 6, no. 2, pp. 442–449, May 1991.
- N. Rasmussen, “Electrical Efficiency Measurement for Data Centers,” p. 19.
- J. Aslan, K. Mayers, J. G. Koomey, and C. France, “Electricity Intensity of Internet Data Transmission: Untangling the Estimates: Electricity Intensity of Data Transmission,” *Journal of Industrial Ecology*, vol. 22, no. 4, pp. 785–798, Aug. 2018.
- L. Hadsell and H. Shawky, “Electricity Price Volatility and the Marginal Cost of Congestion: An Empirical Study of Peak Hours on the NYISO Market, 2001-2004,” *The Energy Journal*, vol. 27, pp. 157–180, Apr. 2006.
- W. Baer, S. Hassell, and B. Vollaard, “Electricity Requirements for a Digital Society,” Jun. 2019.

- T. Heath, B. Diniz, E. V. Carrera, W. Meira, and R. Bianchini, "Energy Conservation in Heterogeneous Server Clusters," in *In Proceedings of the 10th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, 2005.
- H. Ying, S. Greenberg, R. Mahdavi, R. Brown, and W. Tschudi, "Energy Efficiency in Small Server Rooms: Field Surveys and Findings," p. 13, 2014.
- A. Beloglazov and R. Buyya, "Energy Efficient Resource Management in Virtualized Cloud Data Centers," in *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, Melbourne, Australia, 2010, pp. 826–831.
- C. Lefurgy, K. Rajamani, F. Rawson, W. Felter, M. Kistler, and T. Keller, "Energy Management for Commercial Servers," *Computer*, vol. 36, pp. 39–48, Jan. 2004.
- "Energy modeling of two office buildings with data center for green building design - ScienceDirect.":
<https://www.sciencedirect.com/science/article/pii/S037877880700240X>.
- H. Zhang, "Energy Modelling for Data Center Infrastructure," 2017.
- Y. Shang, D. Li, and M. Xu, "Energy-aware Routing in Data Center Network," in *Proceedings of the First ACM SIGCOMM Workshop on Green Networking*, New York, NY, USA, 2010, pp. 1–8.
- Y. Shang, D. Li, and M. Xu, "Energy-aware Routing in Data Center Network," in *Proceedings of the First ACM SIGCOMM Workshop on Green Networking*, New York, NY, USA, 2010, pp. 1–8.
- P. Ranganathan, P. Leech, D. Irwin, J. Chase, and H. Packard, "Ensemble-level Power Management for Dense Blade Servers," p. 12, 2006.
- D. Bouley, "Estimating a data centre's electrical carbon footprint - May 2012 - Schneider Electric South Africa - SA Instrumentation & Control," 2012.
- J. G. Koomey, "ESTIMATING TOTAL POWER CONSUMPTION BY SERVERS IN THE U.S. AND THE WORLD," p. 31.
- D. McIntire, T. Stathopoulos, and W. Kaiser, "etop-Sensor Network Application Energy Profiling on the LEAP2 Platform," in *2007 6th International Symposium on Information Processing in Sensor Networks*, 2007, pp. 576–577.
- D. Economou, S. Rivoire, C. Kozyrakis, P. Ranganathan, and H.-P. Labs, "Full-System Power Analysis and Modeling for Server Environments," p. 8, 2006.
- C. Belady, A. RAWSON, A. JOHN PFLEUGER, and D. TAHIR CADER, "Green grid data center power efficiency metric: PUE and DCIE," Jan. 2008.
- D. Evans, "How the Next Evolution of the Internet Is Changing Everything," p. 11, 2011.
- G. Fettweis and E. Zimmermann, "ICT ENERGY CONSUMPTION – TRENDS AND CHALLENGES," p. 4, 2008.

- R. Hecht-nielsen, “III.3 - Theory of the Backpropagation Neural Network**Based on ‘nonindent’ by Robert Hecht-Nielsen, which appeared in Proceedings of the International Joint Conference on Neural Networks 1, 593–611, June 1989. © 1989 IEEE.,” in *Neural Networks for Perception*, H. Wechsler, Ed. Academic Press, 1992, pp. 65–93.
- D. Roth, “Incidental Supervision: Moving beyond Supervised Learning,” p. 6.
- Y. Joshi and P. Kumar, “Introduction to Data Center Energy Flow and Thermal Management,” *Energy Efficient Thermal Management of Data Centers*, pp. 1–38, Aug. 2013.
- X. Zhu and A. B. Goldberg, *Introduction to Semi-Supervised Learning*, vol. 3. 2009.
- D. Feitosa, R. Alders, A. Ampatzoglou, P. Avgeriou, and E. Nakagawa, “Investigating the effect of design patterns on energy consumption,” *Journal of Software: Evolution and Process*, vol. 29, Feb. 2017.
- I. Journal, “IRJET- Unabridged Review of Supervised Machine Learning Regression and Classification Technique with Practical Task.”
- A. Nouredine, S. Islam, and R. Bashroush, “Jolinar: Analysing the Energy Footprint of Software Applications (Demo),” in *The International Symposium on Software Testing and Analysis*, Saarbrücken, Germany, 2016, p. Pages 445-448.
- S. Han, J. Pool, J. Tran, and W. Dally, “Learning both Weights and Connections for Efficient Neural Network,” presented at the Advances in Neural Information Processing Systems, 2015, pp. 1135–1143.
- D. Held, S. Thrun, and S. Savarese, “Learning to Track at 100 FPS with Deep Regression Networks,” in *Computer Vision – ECCV 2016*, vol. 9905, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 749–765.
- R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, “LIBLINEAR: A Library for Large Linear Classification,” p. 30.
- A. Schneider, G. Hommel, and M. Blettner, “Linear regression analysis: part 14 of a series on evaluation of scientific publications,” *Dtsch Arztebl Int*, vol. 107, no. 44, pp. 776–782, Nov. 2010.
- F. Sebastiani, “Machine learning in automated text categorization,” *ACM Comput. Surv.*, vol. 34, no. 1, pp. 1–47, Mar. 2002.
- K. P. Murphy, *Machine learning: a probabilistic perspective*. Cambridge, MA: MIT Press, 2012.
- S. B. Kotsiantis, I. D. Zaharakis, and P. E. Pintelas, “Machine learning: a review of classification and combining techniques,” *Artif Intell Rev*, vol. 26, no. 3, pp. 159–190, Nov. 2006.

- M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, Jul. 2015.
- Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang, and N. Gautam, "Managing Server Energy and Operational Costs in Hosting Centers," in *Proceedings of the 2005 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, New York, NY, USA, 2005, pp. 303–314.
- Y. Chen, A. Sivasubramaniam, A. Das, Q. Wang, W. Qin, and N. Gautam, "Managing Server Energy and Operational Costs in Hosting Centers," 2005.
- R. Buyya, C. Vecchiola, and S. T. Selvi, *Mastering cloud computing: foundations and applications programming*. Amsterdam ; Boston: Morgan Kaufmann, 2013.
- T. Makris, "Measuring and Analyzing Energy Consumption of the Data Center," 2017.
- "Measuring data center energy consumption in watts per logical image," *SearchDataCenter*. <https://searchdatacenter.techtarget.com/tip/Measuring-data-center-energy-consumption-in-watts-per-logical-image>.
- E. Curry, G. Conway, B. Donnellan, C. Sheridan, and K. Ellis, "Measuring Energy Efficiency Practices in Mature Data Center: A Maturity Model Approach," in *Computer and Information Sciences III*, 2013, pp. 51–61.
- C.-H. Hsu and S. W. Poole, "Measuring Server Energy Proportionality," in *Proceedings of the 6th ACM/SPEC International Conference on Performance Engineering*, New York, NY, USA, 2015, pp. 235–240.
- E. Pakbaznia and M. Pedram, "Minimizing Data Center Cooling and Server Power Costs," in *Proceedings of the 2009 ACM/IEEE International Symposium on Low Power Electronics and Design*, New York, NY, USA, 2009, pp. 145–150.
- L. Rao, X. Liu, L. Xie, and W. Liu, "Minimizing Electricity Cost: Optimization of Distributed Internet Data Centers in a Multi-Electricity-Market Environment," in *2010 Proceedings IEEE INFOCOM*, San Diego, CA, USA, 2010, pp. 1–9.
- C. Finn, P. Abbeel, and S. Levine, "Model-agnostic Meta-learning for Fast Adaptation of Deep Networks," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, Sydney, NSW, Australia, 2017, pp. 1126–1135.
- M. vor dem Berge *et al.*, "Modeling and Simulation of Data Center Energy-efficiency in Coolemall," in *Proceedings of the First International Conference on Energy Efficient Data Centers*, Berlin, Heidelberg, 2012, pp. 25–36.
- T. Zhang, P.-O. Siebers, and U. Aickelin, "Modelling Office Energy Consumption: An Agent Based Approach," 2010.
- S. M. Rivoire, "MODELS AND METRICS FOR ENERGY-EFFICIENT COMPUTER SYSTEMS," p. 172.
- A. Nouredine, R. Rouvoy, and L. Seinturier, "Monitoring Energy Hotspots in Software," *Automated Software Engg.*, vol. 22, no. 3, pp. 291–332, Sep. 2015.

- J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, “Multimodal Deep Learning,” p. 8.
- J. A. S. Benediktsson, “Neural network approaches versus statistical methods in classification of multisource remote sensing data,” *Vancouver, Canada, July 10-14, 1989) IEEE Transactions on Geoscience and Remote Sensing*, Jul. 1990.
- D. West, “Neural network credit scoring models,” *Computers & Operations Research*, vol. 27, no. 11, pp. 1131–1152, Sep. 2000.
- D. Ruta and B. Gabrys, “Neural Network Ensembles for Time Series Prediction,” in *2007 International Joint Conference on Neural Networks*, Orlando, FL, USA, 2007, pp. 1204–1209.
- A. Krogh and J. Vedelsby, “Neural Network Ensembles, Cross Validation, and Active Learning,” in *Advances in Neural Information Processing Systems 7*, G. Tesauero, D. S. Touretzky, and T. K. Leen, Eds. MIT Press, 1995, pp. 231–238.
- I. Sutskever, J. Martens, G. Dahl, and G. Hinton, “On the importance of initialization and momentum in deep learning,” p. 14.
- A. Gandhi, M. Harchol-Balter, R. Das, and C. Lefurgy, “Optimal power allocation in server farms,” in *Proceedings of the eleventh international joint conference on Measurement and modeling of computer systems - SIGMETRICS '09*, Seattle, WA, USA, 2009, p. 157.
- T. D. Sanger, “Optimal unsupervised learning in a single-layer linear feedforward neural network,” *Neural Networks*, vol. 2, no. 6, pp. 459–473, Jan. 1989.
- S. Sra, S. Nowozin, and S. J. Wright, *Optimization for Machine Learning*. MIT Press, 2012.
- W. Vereecken, W. Van Heddeghem, D. Colle, M. Pickavet, and P. Demeester, “Overall ICT footprint and green communication technologies,” in *2010 4th International Symposium on Communications, Control and Signal Processing (ISCCSP)*, Limassol, Cyprus, 2010, pp. 1–6.
- S. Yeo and H.-H. S. Lee, “Peeling the Power Onion of Data Centers,” in *Energy Efficient Thermal Management of Data Centers*, Y. Joshi and P. Kumar, Eds. Boston, MA: Springer US, 2012, pp. 137–168.
- N. Bansal, K. Lahiri, A. Raghunathan, and S. T. Chakradhar, “Power monitors: a framework for system-level power estimation using heterogeneous power models,” in *18th International Conference on VLSI Design held jointly with 4th International Conference on Embedded Systems Design*, 2005, pp. 579–585.
- G. Contreras and M. Martonosi, “Power prediction for Intel XScale/spl reg/ processors using performance monitoring unit events,” in *ISLPED '05. Proceedings of the 2005 International Symposium on Low Power Electronics and Design, 2005.*, 2005, pp. 221–226.

- X. Fan, W.-D. Weber, and L. A. Barroso, "Power Provisioning for a Warehouse-sized Computer," p. 11, 2007.
- N. Horner and I. Azevedo, "Power usage effectiveness in data centers: overloaded and underachieving," *The Electricity Journal*, vol. 29, no. 4, pp. 61–69, May 2016.
- D. Meisner, B. T. Gold, and T. F. Wenisch, "PowerNap: Eliminating Server Idle Power," p. 12.
- M. Hershfield, "Presentation: Myths and Realities About Designing High Availability Data Centers.": <http://blog.morrisonhershfield.com/myths-realities-designing-high-availability-data-centers/>.
- W. K. Hastings and W. K. Hastings, *Printed in Great Britain Monte Carlo sampling methods using Markov*. 2007.
- H. David, E. Gorbato, U. R. Hanebutte, R. Khanna, and C. Le, "RAPL: Memory power estimation and capping," in *2010 ACM/IEEE International Symposium on Low-Power Electronics and Design (ISLPED)*, 2010, pp. 189–194.
- K. Singh and M. Bhadauria, *Real Time Power Estimation and Thread Scheduling via Performance Counters*. 2009.
- M. Shao and R. Futrelle, "Recognition and Classification of Figures in PDF Documents," 2005, pp. 231–242.
- H. Xu and B. Li, "Reducing Electricity Demand Charge for Data Centers with Partial Execution," in *Proceedings of the 5th International Conference on Future Energy Systems*, New York, NY, USA, 2014, pp. 51–61.
- S. Nedeveschi, L. Popa, G. Iannaccone, S. Ratnasamy, and D. Wetherall, "Reducing Network Energy Consumption via Sleeping and Rate-Adaptation," in *NSDI*, 2008.
- R. Tibshirani, "Regression Shrinkage and Selection via the Lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.
- Alliance to Save Energy and Ernest Orlando Lawrence Berkeley National Laboratory, Berkeley, CA (US), "Report to Congress on Server and Data Center Energy Efficiency: Public Law 109-431," LBNL-363E, 929723, Aug. 2007.
- R. Brown, A. to S. Energy, I. C. F. Incorporated, E. R. G. Incorporated, and U. S. E. P. Agency, "Report to Congress on Server and Data Center Energy Efficiency: Public Law 109-431," Jun. 2008.
- E. Steyerberg, T. van der Ploeg, and B. Van Calster, "Risk prediction with machine learning and regression methods," *Biometrical journal. Biometrische Zeitschrift*, vol. 56, Jul. 2014.
- A. W. Lewis, S. Ghosh, and N.-F. Tzeng, "Run-time Energy Consumption Estimation Based on Workload in Server Systems," in *HotPower*, 2008.

- T. Li and L. K. John, "Run-time Modeling and Estimation of Operating System Power Consumption," in *Proceedings of the 2003 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, New York, NY, USA, 2003, pp. 160–171.
- R. Joseph and M. Martonosi, "Run-time power estimation in high performance microprocessors," in *ISLPED'01: Proceedings of the 2001 International Symposium on Low Power Electronics and Design (IEEE Cat. No.01TH8581)*, 2001, pp. 135–140.
- J. Doyle, D. O'Mahony, and R. Shorten, "Server selection for carbon emission control," in *Proceedings of the 2nd ACM SIGCOMM workshop on Green networking - GreenNets '11*, Toronto, Ontario, Canada, 2011, p. 1.
- P. Samuels and M. Gilchrist, *Simple Linear Regression*. 2014.
- P. Ranganathan, "Simulating Complex Enterprise Workloads using Utilization Traces," 2007.
- K. Kersting and S. Natarajan, "Statistical Relational AI : Logic , Probability and Computation," 2011.
- J. E. T. Akinsola, "Supervised Machine Learning Algorithms: Classification and Comparison," *International Journal of Computer Trends and Technology (IJCTT)*, vol. 48, pp. 128–138, Jun. 2017.
- O. F.Y, A. J.E.T, O. Awodele, O. HinmikaiyeJ, O. O. Olakanmi, and J. Akinjobi, "Supervised Machine Learning Algorithms: Classification and Comparison," 2017.
- A. Claassen, H. F. Hamann, M. K. Iyengar, M. P. O'Boyle, M. A. Schappert, and T. G. van Kessel, "Techniques for Analyzing Data Center Energy Utilization Practices," US20080288193A1, 20-Nov-2008.
- K. Makantasis, A. Doulamis, N. Doulamis, and A. Nikitakis, "Tensor-Based Classifiers for Hyperspectral Data Analysis," *IEEE Trans. Geosci. Remote Sensing*, vol. 56, no. 12, pp. 6884–6898, Dec. 2018.
- M. Abadi *et al.*, "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems," *arXiv:1603.04467 [cs]*, Mar. 2016.
- B. Agarwal and N. Mittal, "Text Classification Using Machine Learning Methods-A Survey," in *Advances in Intelligent Systems and Computing*, vol. 236, 2014, pp. 701–709.
- G. Hinton, P. Dayan, B. Frey, and R. Neal, "The 'wake-sleep' algorithm for unsupervised neural networks," *Science*, vol. 268, no. 5214, pp. 1158–1161, 1995.
- L. A. Barroso and U. Hölzle, "The Case for Energy-Proportional Computing," *Computer*, vol. 40, no. 12, pp. 33–37, Dec. 2007.
- Z. Ghahramani, "The EM Algorithm for Mixtures of Factor Analyzers," p. 8.
- G. Verdun *et al.*, "THE GREEN GRID METRICS: DATA CENTER INFRASTRUCTURE EFFICIENCY (DCIE) DETAILED ANALYSIS," p. 16.

H. Singh *et al.*, “The Green Grid Power Sub Work Group”

“The Internet of Things [INFOGRAPHIC],” *blogs@Cisco - Cisco Blogs*:
<https://blogs.cisco.com/diversity/the-internet-of-things-infographic>.

K. G. Brill, “The Invisible Crisis in the Data Center: The Economic Meltdown of Moore’s Law,” p. 8.

G. P. Zhang, “Time series forecasting using a hybrid ARIMA and neural network model,” *Neurocomputing*, vol. 50, pp. 159–175, Jan. 2003.

“Top 10 Energy-Saving Tips for a Greener Data Center.”
<https://www.infotech.com/research/top-10-energy-saving-tips-for-a-greener-data-center>.
[Accessed: 14-Jun-2019].

D. J. Brown and C. Reams, “Toward Energy-efficient Computing,” *Commun. ACM*, vol. 53, no. 3, pp. 50–58, Mar. 2010.

K. Craig-Wood and P. Krause, “Towards the estimation of the energy cost of Internet mediated transactions,” Apr. 2013.

Y. Lv, Y. Duan, W. Kang, Z. Li, and F. Wang, “Traffic Flow Prediction With Big Data: A Deep Learning Approach,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, Apr. 2015.

A. Blum and R. L. Rivest, “Training a 3-Node Neural Network is NP-Complete,” in *Advances in Neural Information Processing Systems 1*, D. S. Touretzky, Ed. Morgan-Kaufmann, 1989, pp. 494–501.

K. Bilal, S. U. R. Malik, S. U. Khan, and A. Y. Zomaya, “Trends and challenges in cloud datacenters,” *IEEE Cloud Comput.*, vol. 1, no. 1, pp. 10–20, May 2014.

M. Avgerinou, P. Bertoldi, and L. Castellazzi, “Trends in Data Centre Energy Consumption under the European Code of Conduct for Data Centre Energy Efficiency,” *Energies*, vol. 10, p. 1470, Sep. 2017.

A. Kumar and P. Yadav, “Unabridged Review of Supervised Machine Learning Regression and Classification Technique with Practical Task,” vol. 05, no. 08, p. 8, 2018.

S. Pelley, D. Meisner, T. F. Wenisch, and J. W. VanGilder, “Understanding and Abstracting Total Data Center Power,” in Proc. Workshop Energy-Efficient Des., 2009, p. 6.

A. Shehabi *et al.*, “United States Data Center Energy Usage Report,” LBNL--1005775, 1372902, Jun. 2016.

P. Dayan, M. Sahani, and G. Deback, “Unsupervised learning,” in *In The MIT Encyclopedia of the Cognitive Sciences*, 1999.

A. Klementiev, D. Roth, and K. Small, “Unsupervised rank aggregation with distance-based models,” in *Proceedings of the 25th international conference on Machine learning - ICML '08*, Helsinki, Finland, 2008, pp. 472–479.

I. Kadayif, T. Chinoda, M. Kandemir, N. Vijaykirsnan, M. J. Irwin, and A. Sivasubramaniam, “vEC: virtual energy counters,” in *Proceedings of the 2001 ACM SIGPLAN-SIGSOFT workshop on Program analysis for software tools and engineering - PASTE '01*, Snowbird, Utah, United States, 2001, pp. 28–31.

Python Documentation: <https://www.python.org/doc/>.

D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio, “Why Does Unsupervised Pre-training Help Deep Learning?,” p. 36.

J. Koomey, “Woldwide electricity use in data centers,” *Environ. Res. Lett*, vol. 24983, pp. 220–236, Jul. 2008.

“Data Center Maturity Model | The Green Grid.” [Online]. Available: <https://www.thegreengrid.org/resources/library-and-tools/239-Data-Center-Maturity-Model->

“ICT Capacity and Utilization Metrics | The Green Grid.” [Online]. Available: <https://www.thegreengrid.org/en/resources/library-and-tools/436-WP#72---ICT-Capacity-and-Utilization-Metrics>

“Virtual Environments and Packages — Python 3.7.3 documentation.” [Online]. Available: <https://docs.python.org/3/tutorial/venv.html>.

APPENDICES

Appendix 1: Code Samples

1 Calculation Model Web Tool

Calculation View

CalculationView.java class represents the view model, responsible for rendering the calculated values in the user interface. As shown below in the code snippet, the class constructs and then holds the view:

```
public class CalculatorView extends Panel implements View {  
1.  
2.     public static final String VIEW_NAME = ((ResourceBundle) VaadinSession.getCurrent().getAttribute("interfaceText")).getString("menulink_calculator");  
3.     public static final String VIEW_URI = "Calculator";  
4.  
5.     private static final double YEARLY_CONSUMPTION_CONSTANT = 8.76;  
6.  
7.     private double electricityUnitCost = 0.12209195;  
8.  
9.     private ComboBox serverEquipmentCb, coolingEquipmentCb, upsEquipmentCb, storageEquipmentCb, networkEquipmentCb, countryCb;  
10.    private VerticalLayout resultsLayout;  
11.  
12.    private TextField serverCountTxt, serverPowerUtilizedTxt, serverPowerIdleTxt, serverUtilizationTxt, coolingCountTxt, coolingPowerTxt, coolingUtilizationTxt  
13.    ,  
14.    upsCountTxt, upsPowerTxt, upsUtilizationTxt, upsEfficiencyTxt, storageCountTxt, diskCountTxt, diskUtilizationTxt, diskPowerTxt,  
15.    networkCountTxt, networkPowerTxt, networkUtilizationTxt, licensingCostTxt, installationCostTxt, itStaffCostTxt, rentalCostTxt, facilityStaffCostTxt;  
16.    public CalculatorView() {  
17.        addStyleName(ValoTheme.PANEL_BORDERLESS);  
18.        setHeight("99%");  
19.  
20.        addStyleName("product-form-wrapper");  
21.  
22.        VerticalLayout root = new VerticalLayout();  
23.        root.setMargin(true);  
24.        setContent(root);  
25.        Responsive.makeResponsive(root);  
26.  
27.        root.addComponent(HeaderBuilder.buildHeader(((ResourceBundle) VaadinSession.getCurrent().getAttribute("interfaceText")).getString("calculator_title")));  
28.  
29.        Component content = buildContent();  
30.        root.addComponent(content);  
31.        root.setExpandRatio(content, 1);  
32.
```

Web Content Building

The web components are responsible for rendering the view for the user. It receives the contents from the model and then transforms it to the view. The code below, first generates HTML components and then binds data to them.

```
1. private Component buildContent() {
2.
3.     VerticalLayout content = new VerticalLayout();
4.     content.addStyleName("dashboard-panels");
5.     Responsive.makeResponsive(content);
6.     content.setSpacing(true);
7.     content.setMargin(true);
8.
9.     try {
10.
11.         resultsLayout = new VerticalLayout();
12.         //resultsLayout.setSpacing(true);
13.         resultsLayout.setSizeFull();
14.
15.         serverEquipmentCb = createComboBox(serverEquipmentCb, "calculator_server_equipme
16.         //serverCountCb = createComboBox(serverCountCb, "calculator_server_coun
17.
18.         coolingEquipmentCb = createComboBox(coolingEquipmentCb, "calculator_cooling_equipme
19.         // coolingCountTxt = createComboBox(coolingCountTxt, "calculator_cooler
20.
21.         upsEquipmentCb = createComboBox(upsEquipmentCb, "calculator_ups_equipme
22.         //upsCountCb = createComboBox(upsCountCb, "calculator_ups_count");
23.
24.         storageEquipmentCb = createComboBox(storageEquipmentCb, "calculator_sto
25.         // storageCountCb = createComboBox(storageCountCb, "calculator_sorage_c
26.
27.         networkEquipmentCb = createComboBox(networkEquipmentCb, "calculator_net
28.         // networkCountCb = createComboBox(networkCountCb, "calculator_network_
29.
30.         countryCb = createComboBox(countryCb, "calculator_country");
31.
32.         // Servers
33.         serverCountTxt = createTextBox(serverCountTxt, "calculator_server_count
34.         serverPowerUtilizedTxt = createTextBox(serverPowerUtilizedTxt, "calcula
35.         serverPowerIdleTxt = createTextBox(serverPowerIdleTxt, "calculator_serv
36.         serverUtilizationTxt = createTextBox(serverUtilizationTxt, "calculator_
```

Calculation Models

As demonstrated in the below code snippets, the models have been encapsulated into Java methods. Those methods take equation variables as method parameters before calculating the power consumption. Calculated values are then returned to the accumulation model:

```
1. public static double calculateServerEnergy(int serverCount, double serverPowerUtilized, double serverPowerIdle, double serverUtilization) {
2.     double anualConsumption = 0;
3.
4.     anualConsumption = (serverPowerUtilized * serverUtilization + serverPowerIdle * (1 - serverUtilization)) * electricityUnitCost * YEARLY_CONSUMPTION_CONSTANT * serverCount;
5.
6.     return anualConsumption;
7.
8. }
9.
10. public static double calculateUpsEnergy(int upsCount, double upsPowerRating, double utilization, double upsEfficiency) {
11.     double anualConsumption = 0;
12.
13.     anualConsumption = (upsPowerRating * utilization / upsEfficiency) * YEARLY_CONSUMPTION_CONSTANT * electricityUnitCost * upsCount;
14.
15.     return anualConsumption;
16. }
17.
18. public static double calculateStorageEnergy(int storageCount, int driveCount, double utilization, double diskPower) {
19.     double anualConsumption = 0;
20.
21.     anualConsumption = diskPower * driveCount * utilization * electricityUnitCost * YEARLY_CONSUMPTION_CONSTANT * storageCount;
22.
23.     return anualConsumption;
24. }
25.
26. public static double calculateCoolingEnergy(int coolingCount, double coolingPower, double coolingUtilization) {
27.     double anualConsumption = 0;
28.
29.     anualConsumption = coolingPower * coolingUtilization * electricityUnitCost * YEARLY_CONSUMPTION_CONSTANT * coolingCount;
30.
31.     return anualConsumption;
32. }
33.
34. public static double calculateNetworkEnergy(int networkCount, double networkPower, double networkUtilization) {
35.     double anualConsumption = 0;
36.
37.     anualConsumption = networkPower * networkUtilization * electricityUnitCost * YEARLY_CONSUMPTION_CONSTANT * networkCount;
38.
39.     return anualConsumption;
40. }
```

2 MALEP

Import additional libraries

```
from sklearn import linear_model

import tkinter as tk

import matplotlib.pyplot as plt

from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg

import pandas as pd
```

Reading the test data from a flat file

```
df = pd.read_csv('server_specs_data.csv')

print('df: ', df)
```

Building dependent and independent variable models

```
X = df[['Processor_Speed', 'Processor_Cores', 'Total_Memory']].astype(float)

Y = df['Energy_Consumption'].astype(float) # output variable (what we are trying
to predict)
```

Regression Model building

```
regr = linear_model.LinearRegression()

regr.fit(X, Y)

print('Intercept: \n', regr.intercept_)
```

```
print('Coefficients: \n', regr.coef_)
```

User Interface Rendering

```
root = tk.Tk()

canvas1 = tk.Canvas(root, width=500, height=300)

canvas1.pack()

# capturing GUI input

Intercept_result = ('Intercept: ', regr.intercept_)

label_Intercept = tk.Label(root, text=Intercept_result, justify='centre')

canvas1.create_window(260, 220, window=label_Intercept)
```

#Streaming data to the UI component

```
Coefficients_result = ('Coefficients: ', regr.coef_)

label_Coefficients = tk.Label(root, text=Coefficients_result, justify='centre')

canvas1.create_window(260, 240, window=label_Coefficients)
```

Capturing Input data for prediction

```
label1 = tk.Label(root, text='Processor Speed: ')

canvas1.create_window(100, 100, window=label1)

entry1 = tk.Entry(root) # create 1st entry box

canvas1.create_window(270, 100, window=entry1)
```

```
label2 = tk.Label(root, text=' Processor Cores: ')

canvas1.create_window(120, 120, window=label2)

entry2 = tk.Entry(root) # create 2nd entry box

canvas1.create_window(270, 120, window=entry2)

label3 = tk.Label(root, text=' Total Memory: ')

canvas1.create_window(140, 140, window=label3)

entry3 = tk.Entry(root) # create 3rd entry box

canvas1.create_window(270, 140, window=entry3)

def values():

    global New_Processor_Speed

    New_Processor_Speed = float(entry1.get())

    global New_Processor_Cores

    New_Processor_Cores = float(entry2.get())

    global New_Total_Memory

    New_Total_Memory = float(entry3.get())
```

```

prediction_result = ('Predicted Energy Consumption: ', regr.predict([[New_Proce
ssor_Speed, New_Processor_Cores, New_Total_Memory]]),

                    ' watts')

label_prediction = tk.Label(root, text=prediction_result, bg='green')

canvas1.create_window(260, 280, window=label_prediction)

button1 = tk.Button(root, text='Predict Energy Consumption', command=values,
                    bg='orange')

canvas1.create_window(290, 170, window=button1)

# plot 1st scatter

figure3 = plt.Figure(figsize=(5, 4), dpi=100)

ax3 = figure3.add_subplot(111)

ax3.scatter(df['Processor_Speed'].astype(float), df['Energy_Consumption'].astype(f
loat), color='r')

scatter3 = FigureCanvasTkAgg(figure3, root)

scatter3.get_tk_widget().pack(side=tk.RIGHT, fill=tk.BOTH)

ax3.legend()

ax3.set_xlabel('Processor Speed ')

ax3.set_title('Processor Speed Vs. Energy Consumption')

# plot 2nd scatter

figure4 = plt.Figure(figsize=(5, 4), dpi=100)

ax4 = figure4.add_subplot(111)

ax4.scatter(df['Processor_Cores'].astype(float), df['Energy_Consumption'].astype(f
loat), color='g')

```



```
scatter4 = FigureCanvasTkAgg(figure4, root)

scatter4.get_tk_widget().pack(side=tk.RIGHT, fill=tk.BOTH)

ax4.legend()

ax4.set_xlabel('Processor_Cores')

ax4.set_title('Processor Cores Vs. Energy Consumption')

# plot 3rd scatter

figure5 = plt.Figure(figsize=(5, 4), dpi=100)

ax5 = figure5.add_subplot(111)

ax5.scatter(df['Total_Memory'].astype(float), df['Energy_Consumption'].astype(float), color='g')

scatter5 = FigureCanvasTkAgg(figure5, root)

scatter5.get_tk_widget().pack(side=tk.RIGHT, fill=tk.BOTH)

ax5.legend()

ax5.set_xlabel('Total_Memory')

ax5.set_title('Total Memory Vs. Energy Consumption')

root.mainloop()
```