Energy Consumption Modelling for 5G Radio Base Stations with Machine Learning

Ringwald, Daniel
danielringwald@outlook.com

Larsson, Daniel
daniel.henrik.larsson@gmail.com

A thesis presented for the degree of
Master of Science

Department of Electrical and Information Technology
Lund University
Supervisor: Fredrik Tufvesson
Examiner: Ove Edfors
May, 2023

Abstract

With an ongoing energy crisis in Europe and an ever-evolving political climate where the environmental impacts of industries are regarded as a top priority, the topic of energy efficiency is high on everyone's agenda. Mathematical optimization of energy consumption requires a model of the problem at hand. In this thesis linear regression is compared with the gradient boosted trees method and a neural network to see how well they are able to predict energy consumption from field data of 5G radio base stations.

The main focus of this thesis has been investigating different architectures of the machine learning models and which features result in the best predictive ability of the models. The models have been trained using field data from deployed radio base station products. This added a complex data processing layer to the work done in this thesis.

The features that resulted in the best performance were, *Number of antennas*, *Configured Max Transmitter Power*, *Frequency*, *Bandwidth*, *IoT capability*, *Product type*, *MAC volume*, *Throughput volume*, *PRB utilization*, *RB symbol utilization* and *MicroSleepTime*.

The findings were that the ANN model had the best predictive ability and it, along with the Gradient Boosted Trees model, performed better than the linear regression model, especially for radio products with more complex configurations. The average errors of the models were $9.47 \cdot 10^{-3}$ EUs for the ANN model, $10.3 \cdot 10^{-3}$ EUs for the Gradient Boosted Trees model and $13.5 \cdot 10^{-3}$ EUs for the linear regression model.

Keywords: Machine learning, energy modelling, radio base station, 5G, neural network, gradient boosted trees

Acknowledgement

We wish to express our gratitude to our supervisors at Ericsson, Jonas Andersson, Niclas Palm, and Pål Frenger for providing us with guidance and support during this work as well as presenting us with challenging problems. We also wish to thank our supervisor at LTH, Fredrik Tufvesson for helping us with keeping us on the right path. Without your help, this thesis would not be possible.

We also wish to extend our gratitude to the whole team at Ericsson for making this work enjoyable and for letting us use the facilities at Ericsson. Finally, we want to thank everyone at Ericsson who offered support and interest in this work, with special mention to Johan Larsson.

Contents

Chapter 1

Introduction

1.1 Background

The topic of energy efficiency is one of utmost importance when discussing how networks are going to be developed in the future. While mobile phone manufacturers have always had the goal of maximizing battery lifetime, the network providers and the companies that supply the network hardware and software have primarily been prioritizing other areas than network energy usage, such as capacity and user performance. The associated costs with power consumption and the resulting CO2 emissions are becoming an increasing concern for network operators [19].

The question is then where the optimizations can be done. Figure 1.1 shows four bar charts that describe what parts make up the operator OPEX (operational expenditure). Figure 1.1(a) shows that 25% of the operator OPEX comes from the network OPEX, from which 90% comes from the energy consumption of the network, see Figure 1.1(b). Figure 1.1(c) then shows that of the energy consumption of the network, 70%-90% comes from the RAN (Radio Access Network) of which 70% of the energy consumption comes from the Radio Base Stations, see Figure 1.1(d). This shows that the energy consumption from Radio Base Stations makes up a significant portion of the Operator OPEX and therefore makes optimization in this area an interesting topic.
As an easy counting example, say that there are 50,000 devices all using on average x watts of energy. If just 1 W is optimized away then the energy savings are in the range of MW. That would result in $50,000 * 24 * 365 = 438$ MWh per year. Using the December average of the "spotpris" (the cost of electricity for the electricity companies) in the south of Sweden in 2022, 248 EUR per MWh, this would save a cost of $248 * 438 = 108,624$ EUR per year [5]. This is just the economic impact. There is also a gain in environmental impact with less energy usage together with an ease of deployment in areas with limited power supply.

As cellular networks get increasingly more advanced, the complexity of the software and hardware increases as more features are implemented. As the set of configurations gets larger the combinations of configurations on a hardware-software product, e.g., a 5G radio base station, increases quickly. As a consequence tractability decreases and optimization becomes harder.

The topic of machine learning (ML) has grown exponentially in the last couple of years and more

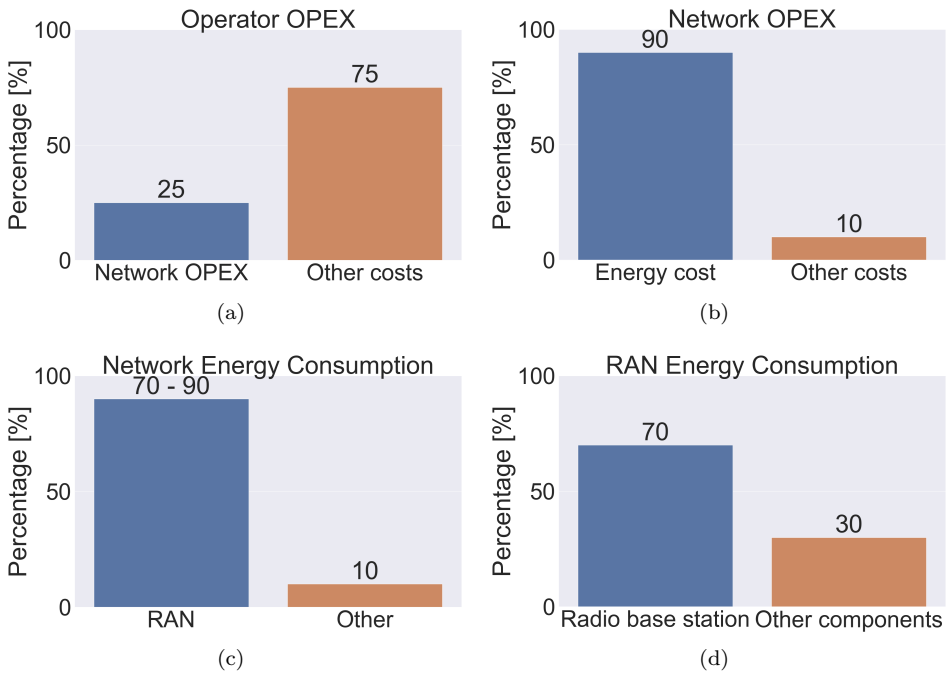


Figure 1.1: The effect of the energy consumption of radio base stations on the operator OPEX [20].

applications are found for the interesting techniques enabled by it. Machine learning has the advantage of not having to fully understand how all the different configurations in software and choices in hardware each impact energy consumption. What can seem like random correlations between configurations and energy consumption can have some complex dependencies that a human might never find.

1.2 Purpose and aims

To optimize energy consumption there needs to be a model in place to evaluate how energy consumption depends on a set of variables. There are many options for modelling, and by using ML, modelling can be performed efficiently on a computer.

The different hardware and software options for 5G radio access network (RAN) offer a host of configurations and performance counters which quickly evolves into a complex web of relations that are very hard to keep track of without sophisticated network analysis which would have to be adapted as new features are implemented. Using ML, the exact relations and dependencies can be left to the algorithm to discover.

This work aims at answering the following questions.

- Can a general model, capable of handling multiple Ericsson radio products be designed?

- How does a general model compare to a more naive approach with separate models for each product?
- Which model has the best performance?
- Which features result in good performance?

1.3 Method

To further develop energy modelling methodology and attempt to answer the questions presented in the previous section, different machine learning algorithm's ability to predict energy consumption is investigated for 5G/4G radio base stations. An artificial neural network (ANN) and gradient boosted trees model will be compared to a more naive approach where linear regression is used to model the energy consumption.

Before training the models and evaluating their performances, a few steps need to be done. These steps include:

- Data extraction
- Feature selection
- Data processing
- Model Architecture

For model evaluation, MSE and MAE will be used as performance metrics. Furthermore, the models will be evaluated visually, see Section 3.10 for more information.

1.4 Previous research

A detailed discussion on applying machine learning in 5G for energy efficiency is shown in [13]. As the complexity of the networks increases the energy consumption rises as well and the need for energy efficiency increases. To aid this transition to a more energy-centered view of networks [13] reviews the state-of-the-art ML applications in energy-efficient 5G networks while also bringing up the difficulties and challenges when dealing with ML tools, such as data acquisition followed by data processing. Furthermore, the trade-off between efficient ML and the simplicity of the models is brought up.

In 2022, a group of researchers at Huawei Technologies in Paris published an article about similar problems to those in this thesis. In [15], researchers performed a related study, modelling the energy consumption of radio products using field data. The results found that by using a neural network with 2 hidden layers it was possible to, on a reasonable level, approximate the energy consumption in Huawei products [15].

In 2022 researchers at Research Institute of China Telecom and Sun Yat-Sen University published an article, in which the building process of an energy consumption model is described. The article

also compares the difference between a traditional energy consumption model and a machine learning energy consumption model [12]. The article chose a neural network as the machine learning energy consumption model. The study was made with data from base stations in the China Telecom network. The results found that by using neural networks to predict the energy consumption of the base stations, and then using Reinforcement Learning to find a strategy selection model, they were able to reduce energy consumption dramatically [12].

The topic of using ML for energy consumption prediction is spread out and is also found in applications such as smart buildings. In [22], machine learning models for regression were used for smart building energy consumption prediction, including Support Vector Regression (SVR), Random Forest (RF), and Artificial Neural Network (ANN). Here it was found that Random Forest performed best out of the tested models with data from residential buildings in a village in Belgium.

In 2021, a group of researchers at the Technical University of Cluj-Napoca, Romania published an article where a comparative analysis of the performance of Random Forest and Gradient Boosting algorithms in the field of forecasting energy consumption based on historical data was made [16]. The data used in the study were gathered from HVAC systems (Heating, Ventilation, and Air Conditioning). The results found that Gradient Boosting was preferable to Random Forest, with Gradient Boosting outperforming Random Forest by 7.9% and 3.8% in terms of the metrics RMSE and MAE, respectively [16].

Chapter 2

Theory

2.1 Telecommunication

2.1.1 5G (NR) and 4G (LTE)

In this thesis, 5G and NR (New Radio) are used interchangeably, as well as 4G and LTE (Long Term Evolution). There are several differences between NR and LTE. One of the biggest differences is the use of new frequency bands for NR. These new frequency bands increase the system bandwidth which results in faster rates of data transfers and lower latency. The new frequency bands are higher than that of LTE which decreases the range of the cells. Signals can also be blocked more easily by obstacles. Although NR can operate on higher frequency bands, it can also operate on the same frequency bands as LTE and thus provide as good coverage as LTE. With the new frequencies and technologies introduced with NR, capacity is increased tenfold in comparison with LTE [1].

A key feature of NR is support for large numbers of antenna elements for both transmission and reception. At the higher frequency bands the primary usage of these antenna elements is for beamforming and at lower frequency bands they enable interference avoidance by spatial separation and full-dimensional MIMO (Multiple Input Multiple Output), which also can be referred to as massive MIMO [4].

2.1.2 Radio Frame

A radio frame in both LTE and NR describes a sequence of data symbols or bits that are transmitted over a radio channel. The duration of a radio frame is set to 10 ms for the subcarrier spacing of 15 kHz. There are other subcarrier spacing configurations supported by NR. These are 30, 60, 120, and 240 kHz. For the 15 kHz subcarrier spacing case, the radio frame can be divided into smaller subframes which typically are 1 ms in length. These can then further be divided into slots and symbols which are 0.5 ms and 1/14 ms respectively. A Physical Resource Block (PRB) is defined as 12 subcarriers and 14 symbols. A summary of the different parts of a radio frame is listed below.

- Radio frame - Time unit of 10 ms
- Subframe - Time unit of 1 ms

- Slot - Time unit of 0.5 ms
- Symbol - Time unit of $\frac{0.5}{7}$ ms
- Subcarrier - Frequency unit of 15 kHz
- Physical Resource Block (PRB) - 12 subcarriers $\times$ 14 symbols
- Resource block symbol (RB symbol) - 12 subcarriers $\times$ 1 symbol

Figure 2.1 shows a diagram of a radio frame with frequency on the y-axis and time on the x-axis. In the figure, one PRB is highlighted as well as one subcarrier with 14 symbols. In LTE, PRBs are allocated to users for transmitting and receiving data. This is due to hard-scheduled reference signals. For NR these reference signals only need to be sent when needed and therefore do not need to be hard-scheduled. This allows data to be sent at the RB symbol level instead of at the PRB level. One RB symbol is also highlighted in the figure.

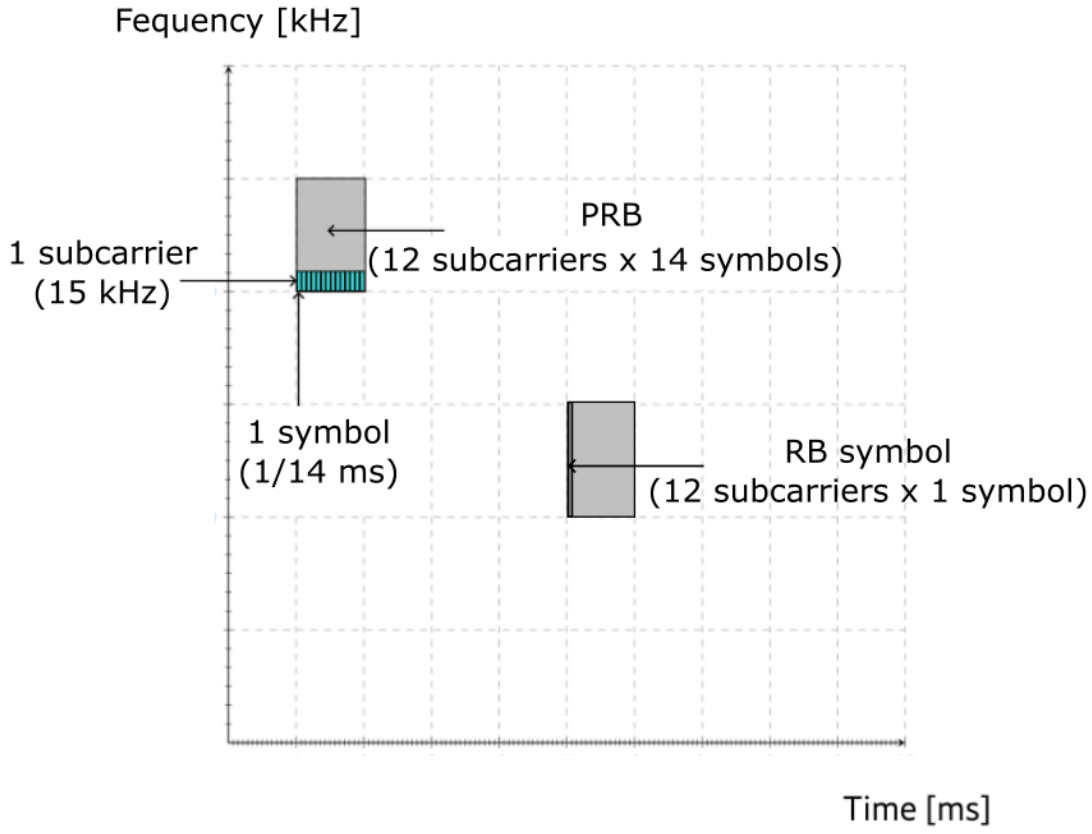


Figure 2.1: A PRB and an RB symbol highlighted in a radio frame.

2.1.3 OFDM

OFDM, short for Orthogonal Frequency-Division Multiplexing, is a method for transmission of information. OFDM is the transmission technique used in the downlink (i.e. from the network to the user terminals) for both LTE and NR. In OFDM the data transmission targeting each user can be mapped to several subcarriers in a very flexible way. This is done in such a way that QAM (Quadrature Amplitude Modulation) symbols are mapped to the orthogonal subcarriers with information about the amplitude and the phase. An example that shows how this mapping can be done with 16-QAM signal constellation points can be seen in Figure 2.2. Using IFFT (Inverse Fast Fourier Transform), the signals in the frequency domain are then sent as sums of sinusoids in the time domain. To circumvent the problem of interference between the symbols a method called cyclic-prefix insertion is used where the end part of the symbol is copied and inserted at the beginning of the symbol. The steps involved in this can be seen in Figure 2.3. In the figure, M is the number of orthogonal subcarriers in the system and N is the size of the IFFT. T_u is the useful symbol duration and T_{CP} is the duration of the cyclic-prefix. To perform coherent demodulation of the signals, an estimation of the channel is needed. This is also shown in Figure 2.3. H_m and n_m represents the complex channel response on subcarrier m and the thermal noise on subcarrier m, respectively. H_m needs to be estimated for each subcarrier m. When the signal is transmitted and then received, FFT (Fast Fourier Transform) can be used to transform the signal back into the frequency domain, where the QAM symbols can be recovered. In OFDM, since the subcarriers are orthogonal they will not interfere with each other and thus several bits can be sent at the same time in parallel. The result is that each subcarrier can send information at a lower data rate which makes it easier to mitigate effects like inter-symbol interference. The overall data rate can be high while the data rate on each subcarrier is kept low [9].

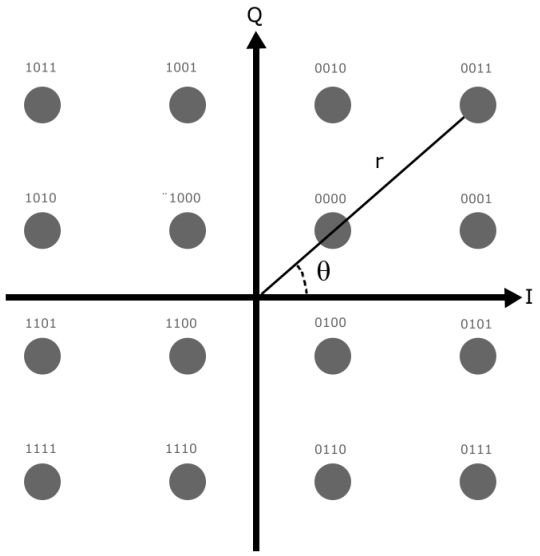


Figure 2.2: 16-QAM where the 4-bit information is mapped to a complex number with phase θ and amplitude r.

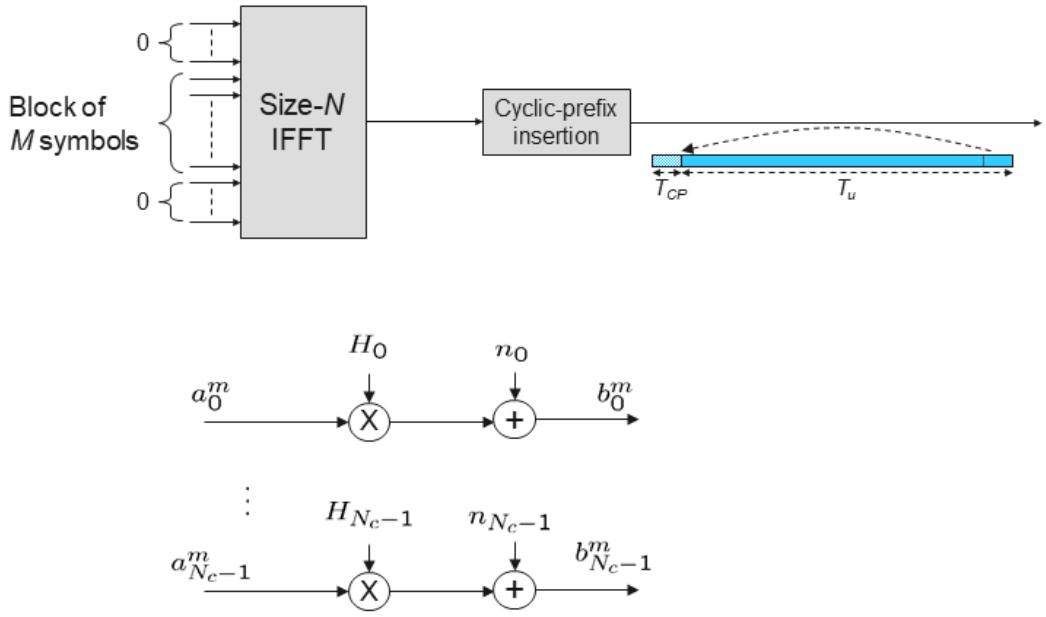


Figure 2.3: OFDM [6]

2.1.4 MIMO

MIMO, or Multiple Input Multiple Output, is the idea of using multiple antennas for data transmission and data reception. There are a number of different advantages when using MIMO technology. Figure 2.4 shows some of the useful features that are available in MIMO. Below are short explanations of the features which are presented in the figure.

Diversity
Diversity achieves higher data reliability by sending multiple copies through multiple antennas. Wireless communications usually suffer from fading effects and by sending multiple independent signals the chance that all are faded drops dramatically.
Beam-forming
This technology allows the antennas to use phase shifts to point the signal in a direction. This allows for constructive interference in some directions and destructive in others. This achieves a higher signal-to-noise ratio and the ability to focus the signals to a position.
SDMA
Using space division multiple access, or SDMA, allows the network cells to reuse frequencies and connect to different users by spatially separating the beams.
Multi-layer transmission
To achieve higher data rates, the possibility of sending multiple signals to a user with multiple antennas is done by sending the data simultaneously across different data streams, or *layers*. To

distinguish between the layers, the user equipment (UE) at the receiving end must have the same number of antennas as there are *layers*.

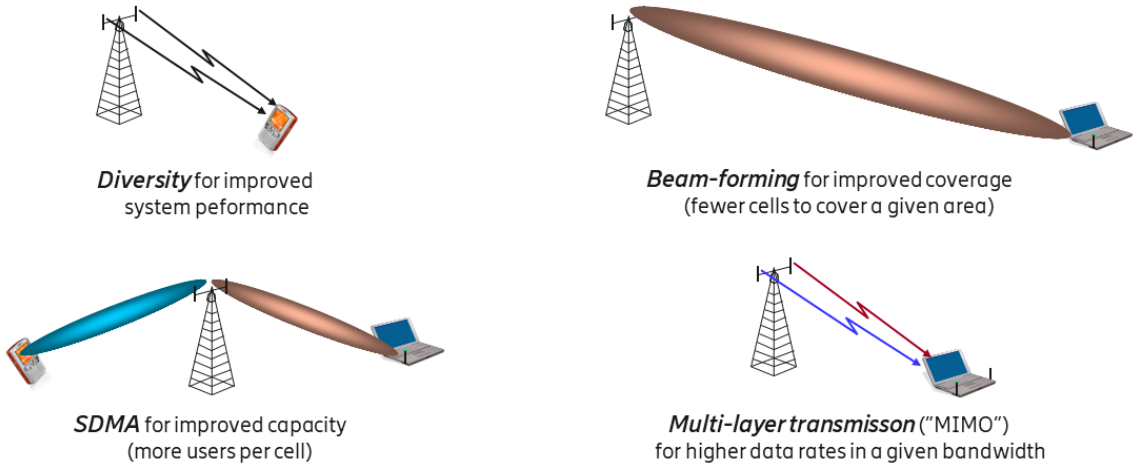


Figure 2.4: Examples of features available with MIMO [6].

2.1.5 Radio base station

A radio base station consists of several different components, which can differ slightly but, typically, it includes an antenna unit, a radio unit, and a baseband unit. The antenna unit is responsible for emitting and receiving radio signals and is often placed on top of the base station tower to ensure as good coverage as possible. The radio unit is used to convert signals between digital and analog form. For transmission it converts a digital signal to an analog signal and amplifies it, for receiving it converts an analog signal to a digital signal. The baseband unit is where the most of the higher layer processing of the signals is done.

Associated with radio base stations are "cells", which are defined as the specific geographical area where the antennas provide coverage.

2.1.6 AIR and Radio products

In this thesis, a number of Ericsson products are analyzed. These can simplistically be classified into two product categories, the AIR (Antenna Integrated Radio) products, and the RADIO products. The AIR products have, as its name implies, the antenna unit integrated into the radio unit. These are 5G products that use massive input massive output-technology, which requires a lot of coaxial cables. Because of this, the antennas are integrated with the radio. This becomes a physical constraint for the product. The RADIO products have separate antennas connected with coaxial cables.

2.1.7 Telecommunication concepts

In this section, a short explanation of telecommunication concepts that are relevant in this thesis is presented.

Uplink/Downlink

In radio communication, there are mainly two directions of communication, one from a user equipment (UE) to a radio base station and one from a radio base station to a UE. These are referred to as the uplink and downlink respectively. These are important to consider when for example looking at bandwidth and frequency as they can differ from each other.

Transmitter/Receiver

The transmitter and the receiver are closely related to the concept of uplink and downlink. If one first considers downlink communication, the transmitter is located in the radio base station, and the receiver is in the UE. For uplink communication, the roles are reversed and the transmitter is located in the UE and the receiver in the radio base station. When discussing transmitter and receiver antennas this means that antennas operating in the uplink are referred to as receiver antennas and antennas operating in the downlink as transmitter antennas.

IoT

IoT stands for Internet of Things and is a collective name that refers to physical objects that have built-in technologies for the purpose of connecting the object to other devices or systems over the internet and exchanging data [2].

In LTE there is a standard extension that is developed to support low-cost IoT devices known as Narrow-band IoT (NB-IoT). The presence of NB-IoT on an LTE carrier is important in this study since it has a large impact on the possibility for the radio base station to enter a sleep mode with reduced energy consumption when traffic is low. LTE cells without an NB-IoT carrier defined will be able to enter sleep mode more often and for longer durations during low traffic hours, and hence consume less energy.

Medium Access Control (MAC) volume

In the Open Systems Interconnection (OSI) model, the MAC sublayer is in the Data Link Layer. This is the layer that is closest to the physical layer, thus it is a good place to measure the amount of data that is flowing through the antennas. MAC volume refers to the data volume measured in the MAC layer.
In this work the following PM (Performance Measurement) level counters were used.

- pmMacVolDl: Aggregated volume of successfully transmitted downlink data exchanges for NR.
- pmMacVolUl: Aggregated volume of successfully transmitted uplink data exchanges for NR.
- pmRadioThpVolDl: Successfully transferred downlink data volume on MAC level for LTE.
- pmRadioThpVolUl: Successfully transferred uplink data volume on MAC level for LTE.

FDD/TDD

Frequency-division duplexing (FDD) and time-division duplexing (TDD) are two methods of separating the uplink and the downlink in a telecommunication system. FDD distributes a part of the allocated frequencies to the uplink and the other part to the downlink. Thus uplink and downlink will not interfere with each other and they are seen as different bands. TDD instead allocates different time slots to the uplink and the downlink. The downlink is usually more resource intensive and because of this more time slots are usually allocated to the downlink than the uplink.

Resource block symbol utilization (NR unique)

Resource block symbol utilization (RB symbol utilization) is a measure of the utilization of radio resources. It measures the fraction of how many RB symbols are used in a cell out of available RB symbols. This is unique for NR for the reasons mentioned in Section 2.1.2.

Physical resource block utilization (LTE unique)

Physical resource block utilization is a measure comparable to RB symbol utilization but instead measures the fraction of how many PRBs are used in a cell out of available PRBs. This is unique for LTE for the reasons mentioned in Section 2.1.2.

Microsleeptime

Micro Sleep Time, found in the counter "$pmMicroTxSleepTime$", is a measurement of how long the power saving feature Micro Sleep Tx is activated. Micro Sleep Tx is a feature of the product which turns off different parts of the hardware when idle, such as power amplifiers, etc. For NR this feature is found under the MO object $NRSectorCarrier$ while for LTE it is found under $SectorCarrier$.

2.1.8 RDI

The data used in this study consist of a configuration management (CM) data set and a performance management (PM) data set. The CM data contains all parameters that are used to configure each radio base station in the network, including configured power, bandwidth, frequency, number of antennas, position, activated energy saving features, etc. The PM data consist of counters that monitor the performance in the network such as the number of connected users, data volume and data rate, consumed energy, etc. There are several thousand CM and PM data entries that are organized in a hierarchical managed object model (MOM). The PM data is collected every 15 minutes and the CM data is (in this study) collected once per day.

Radio data insights (RDI) is a database where data from Ericsson products in the field are stored. This database contains a large amount of data that, with an export interface, can be extracted and analyzed. In this project, the export interface RDI-RAFT is used which is an internal extraction library written in Python. In Figure 2.5 an illustration of the steps included in the data extraction can be seen.

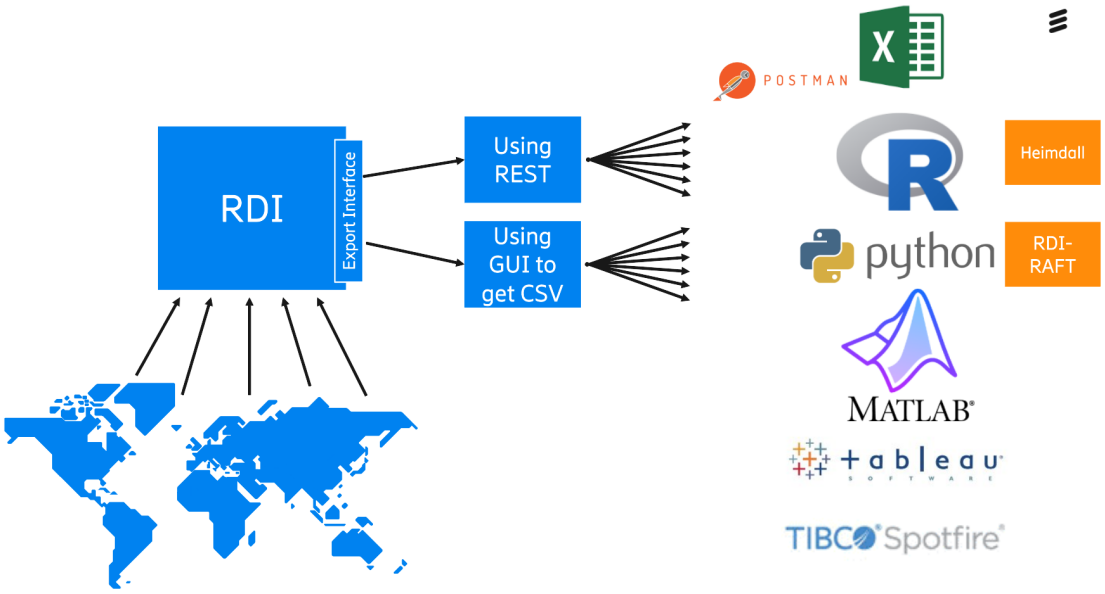


Figure 2.5: Illustration of how the available data from the RDI is gathered.

2.2 Machine Learning (ML)

2.2.1 Overview of ML

The popularization of the term "machine learning" is credited to the researcher Arthur Samuel in his paper "Some Studies in Machine Learning Using the Game of Checkers". There Samuel states that a computer can "learn to play a better game (...) than can be played by the person who wrote the program" [17] while only having access to the rule of the game and some parameters that are incomplete. The idea of machine learning is to build computer programs that learn from experience, i.e. by showing the program more examples it should more accurately make some decisions in a better way.

2.2.2 Linear Regression

Linear regression is a method that tries to approximate scalar values using one or more, so-called, explanatory variables.

The mathematical formulation can be seen in Equation 2.1.

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p + \epsilon \tag{2.1}$$

This equation states how a scalar observation y is dependent on a set of p-dimensional explanatory variables $\{x_1, x_2, ..., x_p\}$ with some additive noise ϵ that acts as a perturbation. It is usually denoted as n number of these equations which can be denoted using matrix notation as in Equation 2.2.

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip} + \epsilon_i = \boldsymbol{x_i}^T \boldsymbol{\beta} + \epsilon_i \ , \ i = 1, ..., n \tag{2.2}$$

The goal of the linear regression is to find a choice of weights $\boldsymbol{\beta}$ where the error is minimized, meaning that a "line", in p dimensions, should be found that minimizes the loss function. The loss function usually used is the mean square error.

Even though the linear regression model is, as the name implies, linear, there are simple manipulations to be done to be able to approximate higher-order polynomials. Modifying Equation 2.1 to Equation 2.3.

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + ... + \beta_p x_{p+1} + \epsilon \tag{2.3}$$

A new variable x_1^2 has been added, which is just the square of the variable x_1. With this simple modification, the linear regression model can now approximate polynomials of order 2. This method can be used to approximate even higher-order polynomials.

2.2.3 Decision Tree

Decision trees are a class of supervised machine learning methods. It works by recursively splitting the training data set into smaller sets until a stopping criterion is met or the data can not be split up more. It works for both classification problems and regression problems.

With only one input variable and one target variable, a decision tree is trained to split the data into subsets, based on the input variable, that provide the smallest loss when comparing predictions to target values. The predictions of the decision tree are the average of the subset for which leaf one ends up in. With many input variables, the root of the tree is decided on which input parameter provides the best split. A decision tree is able to give strong predictions but it can be prone to overfitting in which case ensemble methods can be of use. In Figure 2.6 an example of a simple decision tree can be seen. The blue nodes represent the splits of the subsets and the yellow nodes represent the root node and the leaves. With data points in the range $[2, 10]$ and the function $f(x) = x$ this decision tree has found that the best splits are done $x = 5$, $x = 3$, and $x = 7$.

2.2.4 Gradient Boosting

Gradient boosting or gradient boosted trees is an ensemble method that can be used for both classification and regression. It works by combining many decision trees recursively to create a model with strong predictive ability. The first iteration starts with a single decision tree created using the training data set. Predictions using this model are used and compared to the target values with a loss function. The gradient of the loss function is then used as a target variable to fit a new decision tree. The predicted values for the model are then updated with the predictions of the new decision tree. This process is repeated until a stopping criterion is met. The process is further explained below.

The initial step is to generate a prediction using a weak learner, for example, a decision tree with only one node. Let this prediction be denoted as $\hat{y}_0$. The second step is to calculate the first

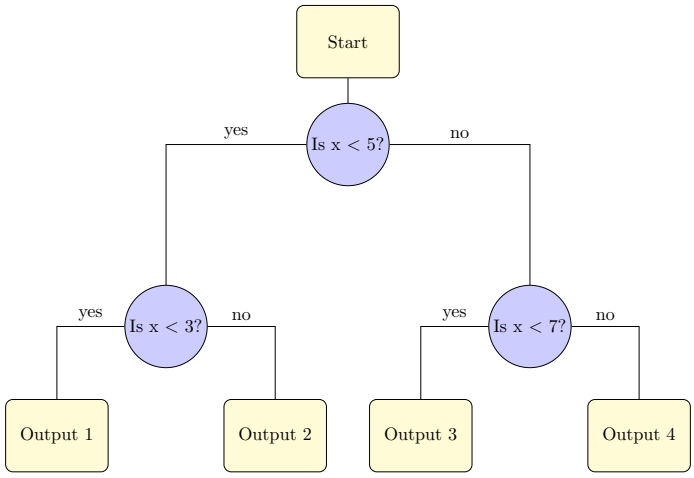


Figure 2.6: Example of a simple decision tree for regression. Root node (start node) and leaves (output nodes) are in yellow and decision nodes are in blue.

residuals r_0 as;

$$r_0 = \frac{\partial L(y, \hat{y}_0)}{\partial \hat{y}_0} \tag{2.4}$$

where y is the true target value.

With this residual a new decision tree is fitted using the input training data set and r_0 as the target. Let the predictions of this tree be denoted as $f_1(x)$.

With the prediction $f_1(x)$ and $\hat{y}_0$ the predictions for the true target variable is updated as;

$$\hat{y}_1 = \hat{y}_0 + \eta f_1(x) \tag{2.5}$$

where η is the learning rate. This process is repeated until a maximum number of trees is reached or a stopping criterion is met.

Generalization

To combat overfitting and generalize the model one can increase the amount of training data or add regularization to the training. One such regularization method in Gradient boosting is called *subsample*. This parameter lies in the range $[0, 1]$ and specifies the fraction of samples to be used for fitting each of the weak learners in the model. Setting the parameter to be less than one, leads to a reduction in variance and an increase in bias and thus reduces overfitting.

2.2.5 Artificial Neural Network

Artificial neural networks (ANNs) were first designed with inspiration from biological neural networks that constitute animal brains [23]. The idea is that you are able to create complex systems

by using simple signals similar to that of the neuron. In an artificial neural network, one of the simplest ways of achieving this is with the perceptron. A perceptron takes the inputs from other nodes in a previous layer and outputs the weighted sum [8]. Combining many of these perceptrons the whole system is able to solve complex problems.

Universal approximation theorem

The possibility of using neural networks to approximate functions was proven by the universal approximation theorem. This theorem states that with no constraints on weights or number of nodes, a one-layer neural network can approximate any function. [11] In theory this is very intriguing because of the possibility of a universal approximator, however, in practice, this does not work. The theorem only guarantees that a solution exists but to find it will be tedious and possibly extremely computationally expensive. In response, the method of adding complexity to the network has the possibility to increase performance while keeping the model computationally feasible, i.e. for a feedforward neural network, this would entail an increasing number of neurons in the layers, or adding extra hidden layers.

Feedforward neural networks

A feedforward neural network is a subset of artificial neural networks where node connections do not form a loop. This means that the information flows in one direction between the layers. There can be many types of feedforward networks but they typically have an input layer, possibly one or more hidden layers, and an output layer [18]. Figure 2.7 illustrates an example of a relatively simple feedforward neural network. The n input nodes are denoted as x, the m hidden nodes in each of the 3 hidden layers are denoted as h and the k output nodes are denoted as y.

Multi Layer Perceptron

A multi-layer perceptron (MLP) is a fully connected feedforward artificial neural network. They consist of three types of layers, an input layer, at least one hidden layer, and an output layer. The nodes in each layer have connections to all other adjacent layers. For example, if an MLP has an input layer of size N, then for each node in the first hidden layer, the output would be;

$$\phi_j([w_1, w_2, ..., w_N], [x_1, x_2, ..., x_N]) = \phi_j(\sum_{i=1}^{N} w_i x_i) \tag{2.6}$$

where $[w_1, w_2, ..., w_N]$ and $[x_1, x_2, ..., x_N]$ are the weights and inputs respectively and j is the node number in the hidden layer. ϕ_j is the activation function defined for node j.

Activation functions

The choice of activation function depends on the problem which is to be solved. Some common activation functions are listed below.

- Linear, $f(x) = x$
- Sigmoid, $f(x) = \frac{1}{1+e^{-x}}$
- Rectified Linear Unit (ReLU), $f(x) = \max(0, x)$

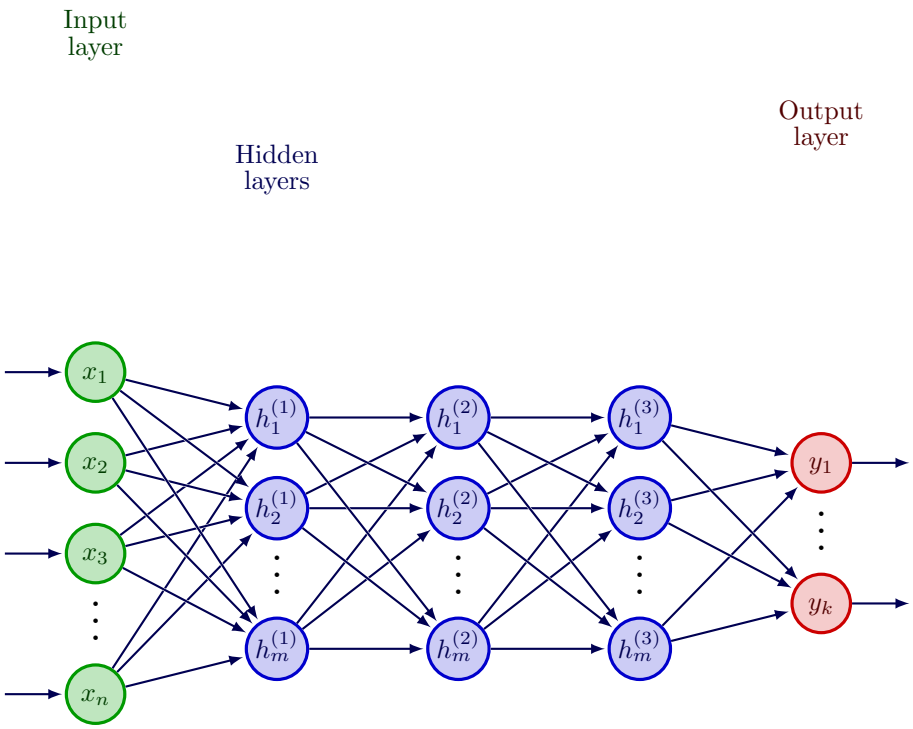


Figure 2.7: Example of a neural network with one input layer, three hidden layers, and one output layer.

With non-linear activation functions, an MLP is able to solve non-linear problems because of the Universal Approximation Theorem.

Optimizers

Stochastic Gradient Descent (SGD) is a popular optimization algorithm for training MLP regressors. The goal of SGD is to minimize the mean squared error (MSE) between the predicted and true values of the target variable. The algorithm works by iteratively updating the model weights.

The update rule for SGD can be expressed as follows:

$$w_{t+1} = w_t - \eta \nabla L(w_t)$$

where w_t is the weight at time step t, η is the learning rate, $\nabla L(w_t)$ is the gradient of the loss function with respect to the weights at time step t. The gradient is estimated using a small random subset (or " batch") of the training data, which helps to reduce the computational cost of the optimization process.

The basic idea behind SGD is to take small steps in the direction of the negative gradient of the loss function, which corresponds to the direction of the steepest descent. By doing so iteratively, the algorithm can converge to a local minimum of the loss function, which possibly corresponds to a good set of weights for the MLP regressor. One potential issue that can occur with SGD is that it can get stuck in local minima, and may not find the global minimum of the loss function [3].

Adam optimizer is an extension of stochastic gradient descent that uses adaptive learning rates and momentum to update the model weights during training. For an MLP regressor, the goal is to minimize the MSE between the predicted and true values of the target variable. The Adam optimizer achieves this by computing an adaptive learning rate for each weight based on the gradients and historical momentum.

The update rule for the Adam optimizer can be expressed as follows:

$$w_{t+1} = w_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t \tag{2.7}$$

where w_t is the weight at time step t, η is the learning rate, ϵ is a small constant to prevent division by zero, $\hat{m}_t$ is the estimate of the first moment (i.e., the mean) of the gradients at time step t, and $\hat{v}_t$ is the estimate of the second moment (i.e., the uncentered variance) of the gradients at time step t. The estimates of the moments are computed using exponential moving averages as follows:

$$\hat{m}_t = \beta_1 m_{t-1} + (1 - \beta_1) \nabla L(w_t) \tag{2.8}$$

$$\hat{v}_t = \beta_2 v_{t-1} + (1 - \beta_2) \nabla L(w_t)^2 \tag{2.9}$$

where $\nabla L(w_t)$ is the gradient of the loss function with respect to the weights at time step t, m_{t-1} and v_{t-1} are the estimates of the moments at the previous time step, and β_1 and β_2 are hyperparameters that control the decay rates of the moving averages.

The Adam optimizer has been shown to be effective for training MLP regressors, as it can adapt the learning rate for each weight based on the gradient history, and in contrast to SGD can also account for momentum effects. This can lead to faster convergence and improved performance on the regression task [10].

Loss functions

Mean squared error (MSE) is calculated as follows:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

where n is the number of observations, y_i is the true value of the ith observation, and $\hat{y}_i$ is the predicted value of the ith observation.

Mean absolute error (MAE) is calculated as follows:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

where n is the number of observations, y_i is the true value of the ith observation, and $\hat{y}_i$ is the predicted value of the ith observation. The absolute value of the difference between the true and predicted values is taken for each observation, and the mean of these absolute differences is calculated to obtain the MAE.

Backpropagation

Backpropagation is an algorithm that is widely used for feedforward neural networks. When fitting the neural network the algorithm calculates the gradient of the loss functions and the term backpropagation strictly refers to this particular step, not how the gradient is used. Backpropagation first calculates the gradient of the loss function with respect to the weights in the final layer of the model and then goes backward in the layers and calculates the gradient iteratively. This works efficiently because the chain rule and calculations do not need to be remade.

Let $w_{ij}^{(l)}$ denote the weights in the network, where l is the layer index, i is the index of the node in the previous layer, and j is the index of the node in the current layer. The gradient of the loss function (L) with respect to these weights can be written as, $\frac{\partial L}{\partial w_{ij}^{(l)}}$.
Using the chain rule, we can express this derivative as:

$$\frac{\partial L}{\partial w_{ij}^{(l)}} = \frac{\partial L}{\partial y^{(i)}}\frac{\partial y^{(i)}}{\partial z_j^{(l)}}\frac{\partial z_j^{(l)}}{\partial w_{ij}^{(l)}}$$

where $z_j^{(l)}$ is the input to neuron j in layer l. The term $\frac{\partial L}{\partial y^{(i)}}$ is the gradient of the loss function with respect to a single output node which can be calculated easily. The term $\frac{\partial y^{(i)}}{\partial z_j^{(l)}}$ is the gradient of the output of node i with respect to the inputs which can be computed using the derivative of the activation function ϕ for layer l:

$$\frac{\partial y^{(i)}}{\partial z_j^{(l)}} = \frac{\partial \phi(z_j^{(l)})}{\partial z_j^{(l)}} = \phi'(z_j^{(l)})$$

For the final term, let $z_j^{(l)}$ be defined as $z_j^{(l)} = w_{ij}^{(l)}a_i^{(l-1)} + b^{(l)}$, where $a_i^{(l-1)}$ is the output from node i in layer $l-1$. The term can now be written as:

$$\frac{\partial z_j^{(l)}}{\partial w_{ij}^{(l)}} = a_i^{(l-1)}$$

The gradient of the loss function with respect to the weights can now be written as:

$$\frac{\partial L}{\partial w_{ij}^{(l)}} = \frac{\partial L}{\partial y^{(i)}}\phi'(z_j^{(l)})a_i^{(l-1)}$$

To calculate the gradient for all the weights in the network, this expression can be used with backpropagation [14].

Generalization

Overfitting is a common issue that can arise when training a model. The effects of overfitting are usually that a model fails to accurately make predictions on data that it was not trained on. The underlying issue is that the model has extracted noise in the training dataset as if the variation in the noise represents the true structure of the data. To evaluate the performance of the network the data is divided into three datasets, training, validation, and test. The training dataset is used for training the model, the validation dataset for selecting the hyperparameters, and the test dataset to evaluate the model's performance on new unseen data. An overfitted model will perform well on the training dataset but not on new data. To combat overfitting and generalize the model one can increase the amount of data the model is trained on or add regularisation parameters in the training stage of the model. Two such regularizations are node dropout and L2-regularisation.

Node dropout assigns a probability for nodes to be removed during different stages of training.

L2-regularizations work by adding a penalty term to the loss function (L);

$$\hat{L} = L + \lambda \sum_i \sum_j w_{i,j}^2 \tag{2.10}$$

with $\lambda = 0$ the original loss function is gained. Larger values for λ affect the update of the weights such that they become smaller, thus reducing the risk of overfitting.

2.2.6 Encoding categorical variables

In machine learning complications often arise when dealing with categorical variables. There are many types of encoding methods that solve these complications and in this section, two methods are presented.

Ordinal encoding

Ordinal encoding encodes a categorical variable as integers where each value is encoded with a unique integer. For example, if there are 10 categories for a categorical variable then each category is represented as a unique integer in the range $[0, 9]$.

One-hot encoding

For one-hot encoding, the different categories for a categorical variable are represented in a vector with the length of the number of categories. The indices in the vector represent the different categories and if a value belongs to a category then the value of the vector at that index is set to one. Let us take the same example as for ordinal encoding. If a value belongs to category 3 with a categorical variable with 10 categories, then that category will be represented as $[0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]$.

2.2.7 Normalization

Normalization is a common method in statistics and machine learning to make sure each feature is on the same scale. If this is not done a feature with values ranging from for example 1000 to 1010 may falsely have a larger impact on the prediction than for a feature with values ranging from 10 to 20. There are many ways of normalizing features but in this paper standard score normalization and min-max normalization are used.

Standard score normalization

The most common way of normalizing features is standard-score normalization also called Z-score normalization. Below the equation for standard score normalization can be seen.

$$Z = \frac{X - \mu}{\sigma} \tag{2.11}$$

where, Z is the normalized values, X the original values, μ the mean value of the feature and σ is the standard deviation.

Min-max normalization

Another way of normalizing features is min-max normalization. This method ensures that each feature value lies within the range $[0, 1]$. The equation for min-max normalization can be seen below for feature X;

$$X_{scaled} = \frac{X - x_{min}}{x_{max} - x_{min}} \tag{2.12}$$

where, X_{scaled} is the normalized values for feature X at index i, x_{min} the minimum value of the feature and x_{max} the maximum value.

Chapter 3

Methodology

3.1 Overview

In this thesis ML techniques, as described above are used to predict the energy consumption of radio base stations in a mobile telecommunication network. To predict the energy consumption three different models with different configurations are used. The three different methods for the models are described in the theory and are the following:

- Linear regression
- Gradient Boosted Trees
- Artificial Neural Network

In this section, the features chosen to be included in the models are presented. It also includes a description of how the data is structured in the RDI as well as how it was extracted and processed. Finally, the architecture of the models and model training is presented.

3.2 Software

The code was implemented in Jupyter Notebook using Python 3.11.1. For the neural network, TensorFlow 2.12.0-rc0 was used with the Keras API. Keras is an open-source library for interfacing with artificial neural networks with Python. For the linear regression models and for the Gradient Boosted Trees model Scikit-learn 1.2.2 was used. The network analysis was performed by the Python library NetworkX 3.0. Internal Ericsson data export tools were used for the data extraction from an internal database. The open-source library Pandas 2.0.0 was also used.

3.3 Data structure

To organize data for a product, the data is structured according to the Managed Object Model (MOM). The MOM is a software abstraction to organize the different parts in a radio base station. Some are a direct representation of the hardware, e.g. the energy meter, while some are pure

software, e.g. the cells. The result is that to acquire a total dataset, data has to be pulled from different parts of the MOM, i.e. the cell has to be retrieved to extract the data volume.

3.4 Data Extraction

The data that was used is extracted directly from Ericsson's customers, and stored in an internal database called RDI (Radio Data Insight). This database has export tools that were used to find specific data from specific operators, grouped into attributes and classes. The data consists of measurements of the configuration of the products (CM) and performance measurements (PM). Examples of configuration measurements are features such as product name and bandwidth. Examples of performance measurements are energy consumption and PRB utilization.

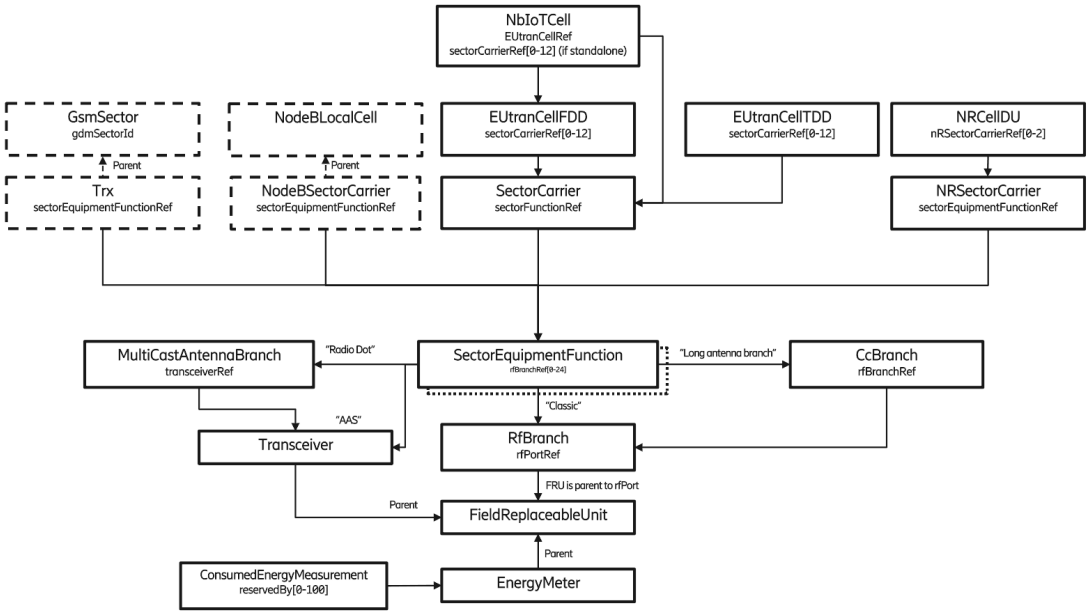


Figure 3.1: Example of connections between classes for a radio unit [21].

The extracted data needs to be formatted in a way suitable for the training of the model and the first step is to find all connections between the data points. Each data point in the extracted dataset has a variable called full descriptive name (fdn). This variable is unique within a network and can be used in reference pointers by other managed objects to indicate a relationship between the objects. The relation between managed objects in the MOM can be derived either from reference pointers to other fdn-names or by knowledge of a fixed relationship between objects (e.g. parent-child). An example of the MO-class object *EnergyMeter* is shown in Figure 3.2.
The map in Figure 3.1 shows the connections between classes for a specific site. To find these connections in the MOM there are a number of different pointers that have to be extracted together

	fdn	rdiTimeStamp	ropCount	pmConsumedEnergy
6376169	SubNetwork=XX,SubNetwork=RAN,SubNetwork=MRAN05...	2023-02-01 02:45:00	1	5.821918
1424077	SubNetwork=XX,SubNetwork=RAN,SubNetwork=MRAN27...	2023-02-01 17:00:00	1	4.280822
3450295	SubNetwork=XX,SubNetwork=RAN,SubNetwork=LRAN01...	2023-02-01 01:30:00	1	4.280822
2474427	SubNetwork=XX,SubNetwork=RAN,SubNetwork=LRAN26...	2023-02-01 01:15:00	1	5.650685
909452	SubNetwork=XX,SubNetwork=RAN,SubNetwork=MRAN21...	2023-02-01 22:45:00	1	9.075342

Figure 3.2: A sample of the pandas data frame containing the information of the object (fdn) with a timestamp of the measurement, the result output period (ropCount) count, and normalized energy consumption (pmConsumedEnergy).

with the data that will go into the model. This is either a pointer that is extracted precisely as stand-alone features, or ones that have to be found from the fdn variable. The pointers are either pointing $from$ or to an object. With the knowledge of this mapping and the different pointers available, one can find the connecting path between, for example, the class EnergyMeter, which has the energy consumption measurement, and the class NRSectorCarrier, which has the bandwidth parameters. These mappings are not always the same in every case and iterating through all paths for every site will be computationally inefficient. Another way is to create a list of edges and vertices where the vertices are classes and the edges are to what other classes that vertex points to. This list of edges and vertices could then be made into an undirected graph which was done using the graph tool NetworkX. From the resulting graph, a set of subgraphs could be extracted where one can see all the groups of connected components. In Figure 3.3, examples of two simple connected components can be seen. In these connected components, one can see which individual measurements that are connected to each other.

An analysis of the subgraphs is done and complex groups like the one seen in Figure 3.4 can be discarded. Complex groups like this with more than two radio units are fairly rare and can be discarded with barely any loss to the amount of data. Groups with more radio units than energy meters, as well as groups with no 5G components, like the group on the right in Figure 3.3, will also be discarded. The groups that are kept can then be used to map the correct measurements to each other and be made into a row in a data frame representing an observation. An illustration of this can be seen in Figure 3.5.

3.5 Chosen features

In this section, the features that were chosen to be included in the models will be presented. Other features were studied but the ones presented here are the ones that gave better performance when evaluating the models.

3.5.1 Configuration parameters

Configuration parameters are parameters that describe the configuration of an object, e.g. a radio unit, in the network. This *configuration data* is taken once a day. The following features are for

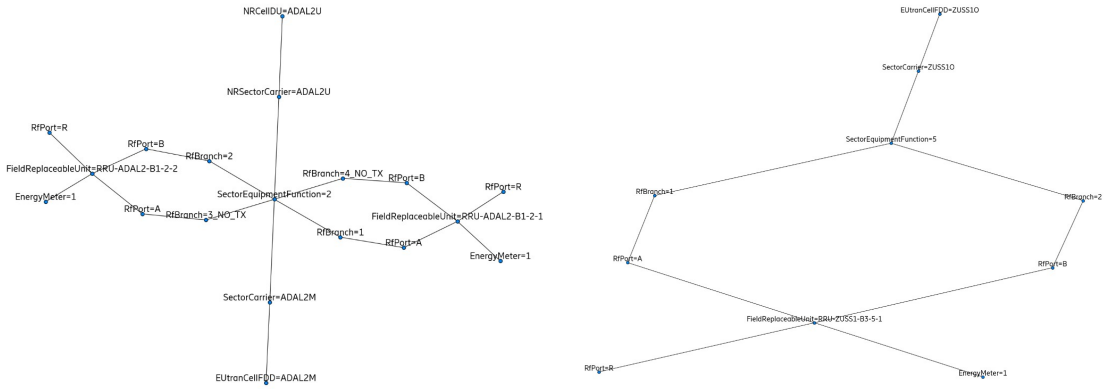


Figure 3.3: Example of two simple connected components found in the graph. The left one has two radio units, one 5G cell, one 4G cell, and two energy meters. The right one has one radio unit, one 4G cell, and one energy meter [21].

the NR cells:

- Number of Transmitter Antennas
- Number of Receiver Antennas
- Configured Max Transmitter Power
- Uplink Frequency
- Downlink Frequency
- Uplink Bandwidth
- Downlink Bandwidth

For LTE cells the features are:

- Number of Transmitter Antennas
- Number of Receiver Antennas
- Configured Max Transmitter Power
- Uplink Frequency
- Downlink Frequency
- Uplink Bandwidth
- Downlink Bandwidth

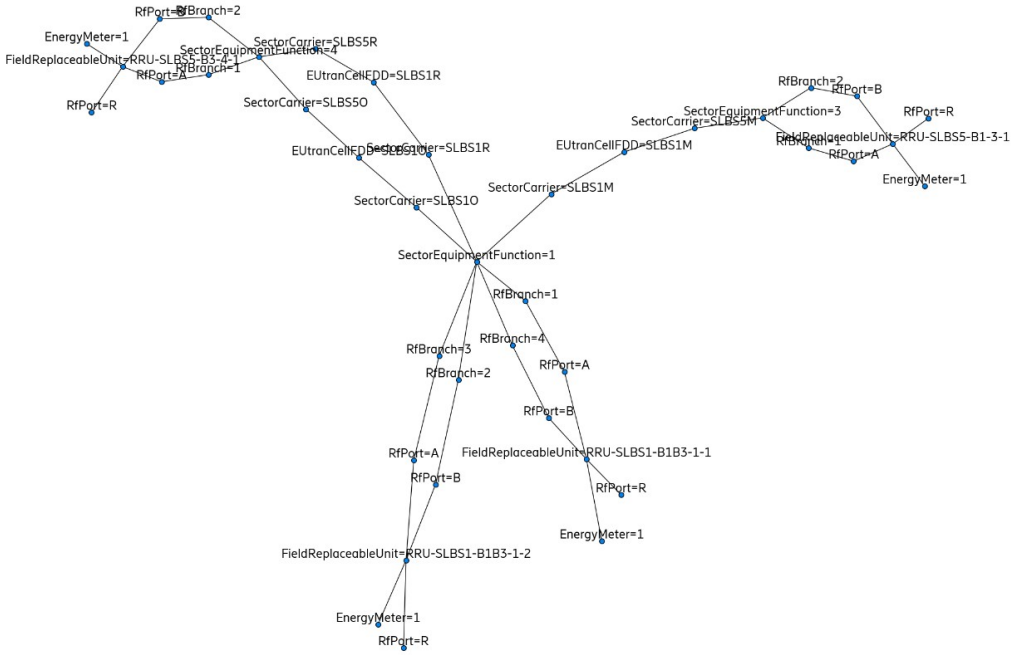


Figure 3.4: Example of a complex connected component found in the graph. This has four radio units and four energy meters [21].

Other configuration parameters are:

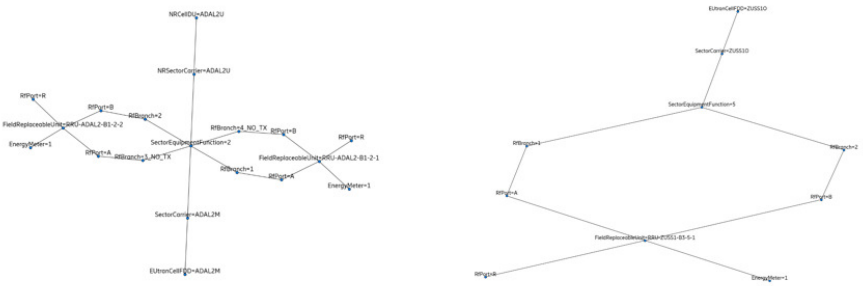
- IoT-cell
- Type of product/Product name

The IoT-cell parameter is a boolean that indicates wheater a radio unit supports IoT. In Table 3.1, all configuration parameters are presented with their CM attribute, which MO-class the attribute belongs to, and the unit of measurement of the parameter.

3.5.2 Performance parameters

Performance parameters are taken once every 15 minutes and are counters that measure things such as data volume and energy consumption aggregated over the 15-minute interval. The performance parameters that are present in this thesis for NR are:

- MAC volume downlink
- MAC volume uplink



CM attributes
PM counters

	Feature 1	Feature 2	...	Feature *K*	Target
Observation 1					
Observation 2					
...					
Observation *N*					

Figure 3.5: Illustration of mapped measurements into a data frame with features as columns and observations as rows [7].

Table 3.1: Unit of measurement for each configuration parameter.

Parameter	CM attribute	MO-class	Unit
# Transmitter antennas (NR/LTE)	noOfUsedTxAntennas	(NR)SectorCarrier	-
# Reciever antennas (NR/LTE)	noOfUsedRxAntennas	(NR)SectorCarrier	-
Configured max TX power (NR/LTE)	configuredMaxTxPower	(NR)SectorCarrier	mW
Uplink/Downlink frequency (LTE)	freqBand	EUtranCellFDD	kHz
Downlink frequency (NR)	frequencyDL	NRSectorCarrier	kHz
Uplink frequency (NR)	frequencyUL	NRSectorCarrier	kHz
Downlink Bandwidth (LTE)	dlChannelBandwidth	EUtranCellFDD	kHz
Downlink Bandwidth (NR)	bSChannelBwDL	NRSectorCarrier	kHz
Number of IoT cells	-	-	-
Product name	productData.productName	FieldReplaceableUnit	-

- RB symbol utilization

For LTE cells the parameters are:

- PRB utilization
- Throughput volume uplink
- Throughput volume downlink
- MicroSleeptime

The target parameter, the energy consumption, is extracted as a performance counter and is not specific to either NR or LTE. In Table 3.2, all performance parameters are presented with their PM attribute, which MO-class the attribute belongs to, and the unit of measurement of the parameter. There are no explicit PM attributes for PRB utilization and RB symbol utilization. Instead, these were calculated using a number of PM attributes. The formula and PM attributes for calculating PRB utilization and RB symbol utilization can be seen in equations 3.1 and 3.2 respectively.

$$\text{PRB utilization} = 100 \times \frac{\text{Total used PRBs}}{\text{pmPrbAvailDl}} \tag{3.1}$$

where,

$$\begin{aligned}\text{Total used PRBs} = {} & \text{pmPrbUsedDlBcch} \\ & + \text{pmPrbUsedDlDtch} \\ & + \text{pmPrbUsedDlPcch} \\ & + \text{pmPrbUsedDlSrbFirstTrans}\end{aligned}$$

$$\text{RB symbol utilization} = 100 \times \frac{\text{Total used RB symbols}}{\text{pmMacRBSymAvailDl}} \tag{3.2}$$

where,

$$\begin{aligned}\text{Total used RB symbols} = {} & \text{pmMacRBSymCsiRs} \\ & + \text{pmMacRBSymUsedPdcchTypeA} \\ & + \text{pmMacRBSymUsedPdcchTypeB} \\ & + \text{pmMacRBSymUsedPdschTypeA} \\ & + \text{pmMacRBSymUsedPdschTypeABroadcasting}\end{aligned}$$

Table 3.2: Unit measurements for each performance counter. (*) denotes that the counter is calculated from multiple other counters as shown in equations 3.1 and 3.2.

Parameter	PM attribute	MO-class	Unit
MAC volume downlink (NR)	pmMacVolDl	NRCellDU	Byte
MAC volume uplink (NR)	pmMacVolUl	NRCellDU	Byte
PRB Utilization* (LTE)	-	EUTranCellFDD	%
RB symbol utilization* (NR)	-	NRCellDU	%
Microsleep time	pmMicroTxSleepTime	SectorCarrier	s
Throughput volume uplink (LTE)	pmRadioThpVolUl	EUTranCellFDD	kbit
Throughput volume downlink (LTE)	pmRadioThpVolDl	EUTranCellFDD	kbit
Energy consumption	pmConsumedEnergy	EnergyMeter	Wh

3.6 Dataset

The datasets that are used are extracted using the RDI-Raft interface, where the features seen in the previous section are selected. All datasets contain the selected features. Below, the different datasets that are used are presented.

Dataset 1

The first dataset is from a single customer on a single weekday during February of 2023. As can be seen in Figure 3.6, the number of products differs wildly. In this set, products that there are deemed to be too few samples of are also removed. The products with fewer samples than 5% of the number of samples for the product with the most samples were removed. This narrows the number of products that the model can handle but removes products that are potentially only used at test sites and other products that number in the single digits.

The numeric values for the data splits for all different products can be seen in Table 3.3.

Dataset 2

To observe different behaviors of the model depending on the dataset, an "even" dataset is used. The dataset is "even" meaning that there is an equal number of samples from all products that were used in Dataset 1. To equalize the dataset, three consecutive days are picked in February 2023, the samples are then grouped together by product. Then the same number of random samples for each product was chosen, where the number of samples is the same as the number of samples of the product with the lowest amount of samples. The data split is visualized in Figure 3.7.

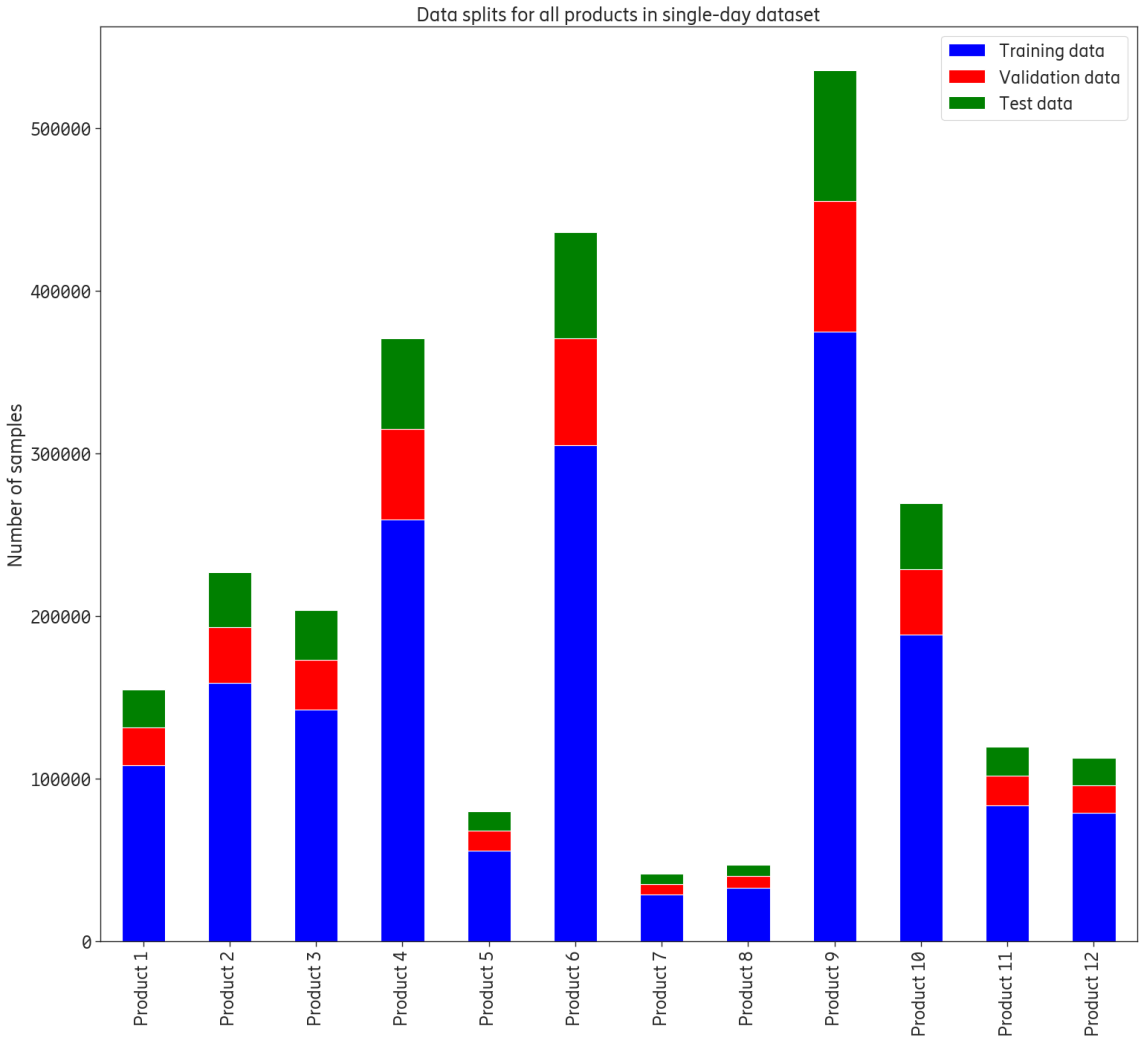


Figure 3.6: Distribution of data split between training, validation, and test data for dataset with field data from a single day.

3.7 Data processing

To handle missing values in the dataset the first step was to remove observations with missing values for all 5G features. This is done for limiting the data to fit the scope of this thesis, where 5G radio base stations are of interest. Otherwise, missing values were replaced with the value zero since, for example, 4G features are present in 5G-only products. Setting those features to zero indicates to the model that the feature is not *relevant*, or "*turned off*". The resulting data frame has to have the same features in the columns which means that all observations require all column fields. In the dataset, products with multiple cells are present. Those observations that have fewer cells than

Product	Training split	Validation split	Test split
Product 1	108308	23209	23210
Product 2	158960	34063	34063
Product 3	142452	30525	30526
Product 4	259527	55613	55614
Product 5	55969	11994	11994
Product 6	305196	65399	65400
Product 7	29037	6222	6223
Product 8	33061	7084	7085
Product 9	374877	80331	80331
Product 10	188618	40418	40419
Product 11	83798	17957	17957
Product 12	78892	16906	16906

Table 3.3: The single day dataset data split.

the maximal number of cells have zeroes in those columns.

The categorical feature "product name" was processed using one-hot encoding.

Samples of energy consumption that were higher than 300 Wh, as they are considered to be outliers when comparing with the rest of the dataset. For some older products there is a known bug that rarely and sporadically can generate unreasonably large energy measurements. For example the measurements can sometimes be 10^{12} Wh for a 15 minutes reporting period.

For improved training of the models, the data was normalized using min-max normalization to make each variable have values in an interval from 0 to 1.

3.8 Model Architecture

The model architectures are, in this section, presented separately for the three different models that are used. The three models are the ANN model, the linear regression model and the Gradient Boosted Trees model.

3.8.1 Linear regression models

The linear regression model is made up of several linear regression model, one for each product. This means that for n different products there are n different linear regression models. The explanatory variables used in the regression models therefore does not include the product name. Otherwise, all features presented in Section 3.5 are used as explanatory variables. To improve the linear regression models, non-linear quadratic and cubic terms are added for the performance parameters PRB Utilization and RB symbol utilization. The equation for the linear regression models then look like the following;

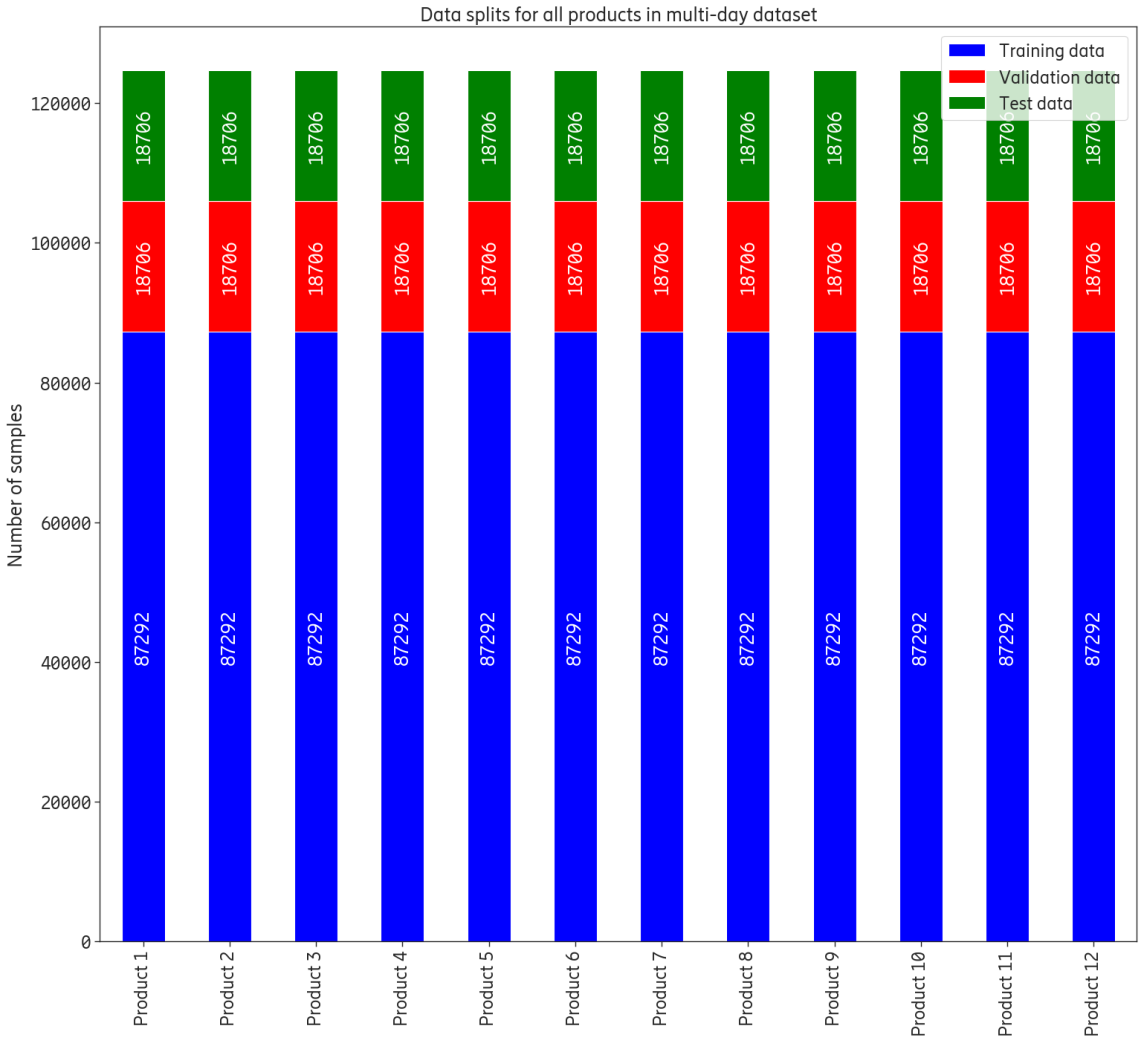


Figure 3.7: Distribution of data split between training, validation, and test data for the dataset with field data from three consecutive days. Equalized dataset with the same number of samples for all products

$$y = \beta_0 + \beta_1 x_{Feature1} + ... + \tag{3.3}$$

$$\beta_{p-5} x_{pmPRBUtil} + \beta_{p-4} x^2_{pmPRBUtil} + \tag{3.4}$$

$$\beta_{p-3} x^3_{pmPRBUtil} + \beta_{p-2} x_{pmRbsymUtil} + \tag{3.5}$$

$$\beta_{p-1} x^2_{pmRbsymUtil} + \beta_p x^3_{pmRbsymUtil} + \epsilon \tag{3.6}$$

where, pmPRBUtil and pmRbsymUtil refers to PRB utilizatin and RB symbol utilization respec-

tively. All other features only have linear terms. The same equation is used for both datasets.

3.8.2 Gradient Boosted Trees model

The architecture of a Gradient Boosted Trees model is built during the training stage of the model. To set up the training, the initial weak learner needs to be defined. In this model, the weak learner is a decision tree containing a single node which's prediction, $\hat{y}_0$, is the average target value in the training data. Figure 3.8 shows how the model architecture will look like after training, beginning with the initial weak learner and then iteratively adding weak learners to create a strong predictor. In this model the weak learners are all decision trees.

The max depth of the weak learners generated during training is set to four. This parameter can be tuned in order to increase performance.

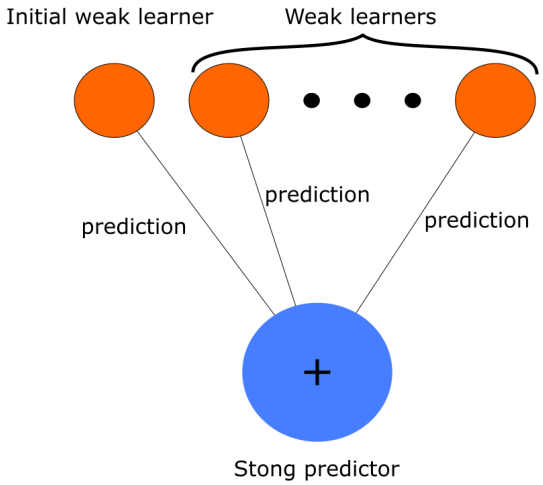


Figure 3.8: Architecture of the Gradient Boosted Trees model.

3.8.3 ANN

The artificial neural network architecture is chosen after iterating through a number of different choices and evaluating the performance. The chosen architecture can be seen in Table 3.4 and a visualization of how the ANN is organized is shown in Figure 3.9.The activation functions between the layers in the ANN are shown in Table 3.5.

3.9 Model training

The dataset is split into three sets, the training set, the validation set, and the test set. The training set contains 70% of the data, the validation set 15%, and the test set 15%. The ANN model and the Gradient Boosted Trees model are trained on the training set and during each step of the training,

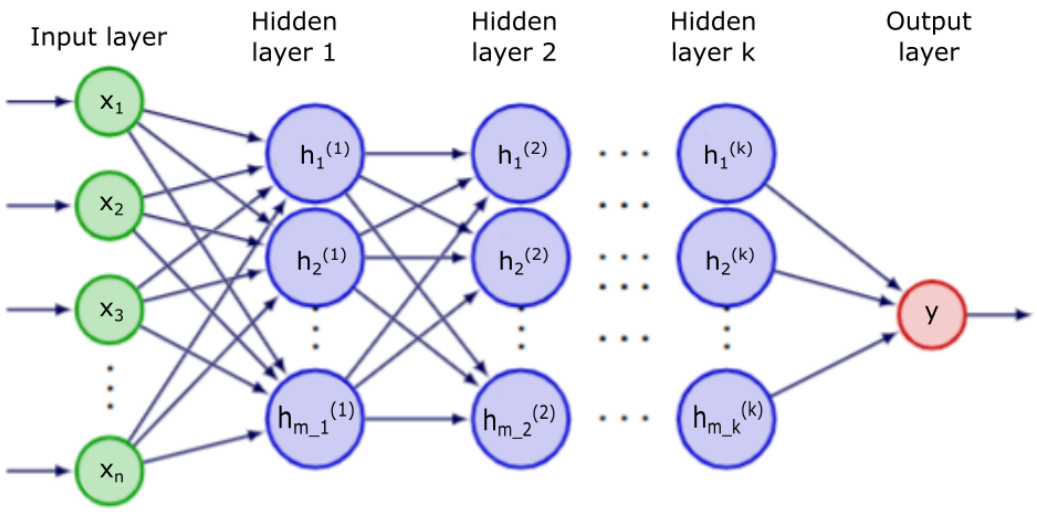


Figure 3.9: Architecture of the ANN model with $k = 7$, $n = 85$ and $\mathbf{m}$ has sizes as in Table 3.4.

Table 3.4: The number of nodes for each layer in the ANN model architecture.

Layer	**#Nodes**
Input	85
Hidden	300
Hidden	300
Hidden	250
Hidden	200
Hidden	150
Hidden	100
Hidden	100
Output	1

Table 3.5: Activation functions between the layers in the artificial neural network that are used in the ANN models.

Layer type	**Activation function**
Hidden	ReLU
Output	linear

the model is evaluated on the validation set. For the ANN model the final weights are chosen as those that give the best score on the validation set. The Gradient Boosted Trees model training

is stopped after 1000 iterations or if the validation loss does not improve for 10 iterations with a tolerance of 10^{-6}.

3.9.1 Hyperparameters

ANN
For the ANN model, the Adam optimizer, see Section 2.2.5, was chosen as the optimization method with learning rate $\eta = 0.01$. The parameters for the Adam optimizer as in equations 2.7, 2.8 and 2.9 were chosen with the default values ($\beta_1 = 0.9$, $\beta = 0.999$, and $\epsilon = 7 \cdot 10^{-7}$) as in Keras, see Section 3.2. The L2-regularization parameter and the node dropout rate was chosen to zero as this showed to result in the best performance. The batch size was chosen as 32 as, again, this resulted in the best performance.

Gradient Boosted Trees
For the Gradient Boosted Trees model, the *subsample* parameter was chosen as one which means that the added weak learners are fitted using the negative gradient of the residuals r (Gradient Descent). Default values for the parameters were chosen, as in Scikit-learn (Section 3.2), with learning rate $\eta = 0.01$.

3.9.2 Loss function

MSE was chosen as the loss function when training model parameters. The main difference between *MAE* and *MSE* is that *MSE* will punish large errors more than *MAE*. This is a desired feature of the loss function.

3.10 Model evaluation

Two metrics are used to evaluate how the model performs. The mean squared error (MSE) and the mean absolute error (MAE). While the models have been trained using MSE as the loss function, the MAE is also of interest since the study is done on energy data, more specifically with Wh as the energy unit. The advantage of using MAE is the gain in interpretability. MSE is a metric that is hard to understand for humans and absolute values of energy make more sense when evaluating the models.

For linear regression, there is one model used for each product. To be able to compare the models with linear regression, the mean of the resulting MSEs and MAEs are used as the comparative number where the predictions and the data are re-normalized with the same normalizing values as for the data that is fed into the ANN and the Gradient Boosted Trees models.

Visually the models are compared using time series. In practice, this is done by calculating the mean of the predictions and target values for each timestamp. By making this comparison, a more tractable way of analyzing the results is achieved where the analysis is not simply done by observing a number in a table. It can also be of interest to investigate how the network at large behaves.

Chapter 4

Results

4.1 Overview

In this section, results attained by the different models are presented for the test set of the different datasets. The results include;

- Total MSE and MAE
- Which product had the best and worst MSE and MAE
- Prediction vs target plots
- Time series plots for different products and models
- Inferences made using the models

Note that the energy consumption is normalized for confidentiality reasons and is presented in units of EU (Energy Unit). For all plots in this section, energy consumption is scaled with magnitude 100. It was hinted before how much an EU is since 300 Wh is considered an outlier. Time series plots for all products and models on Dataset 2 can be seen in figures 6.1, 6.2, and 6.3, in Appendix A. Also, tables showing the MSE and MAE for all products and models on Dataset 2 can be seen in tables 6.1, 6.2 and 6.4, in Appendix A.

4.2 Dataset 1

In this section, statistics of the models trained on Dataset 1 is presented. Table 4.1 shows the total MSE and MAE as well as the number of samples used in the training set. The MSE and MAE for the linear regression models are calculated as the mean MSE and MAE over all products. Both MSE and MAE was lowest for the linear regression model followed by the ANN model and the Gradient Boosted Trees model, in that order. To compare performance on individual products, MSE and MAE was calculated separately for each product. In Table 4.2 and Table 4.3, the results of this is shown. The ANN model performed best for the products with the best MSE and MAE, while the linear regression model performed best on the product with the worst MSE and MAE.

Table 4.1: Total MSE and MAE for the different models using Dataset 1. The best score for each metric is highlighted in bold.

Model	MSE $[(\text{EU})^2 \cdot 10^{-4}]$	MAE $[\text{EU} \cdot 10^{-3}]$	#Samples
ANN	2.25	10.7	1818695
Linear regression	**2.04***	**9.79***	1818695
Gradient Boosted Trees	2.56	11.3	1818695

Table 4.2: Best MSE and MAE for the test data for the different models using Dataset 1. The best score for each metric is highlighted in bold. *#Samples* denotes the number of *training* samples.

Model	MSE $[(\text{EU})^2 \cdot 10^{-4}]$	MAE $[\text{EU} \cdot 10^{-3}]$	Product	#Samples
ANN	**0.609**	**5.24**	Product 8 (MSE) Product 12 (MAE)	33061 (MSE) 78892 (MAE)
Linear regression	0.685	5.26	Product 4 (MSE) Product 12 (MAE)	259527 (MSE) 78892 (MAE)
Gradient Boosted trees	0.712	5.66	Product 1	108308

Table 4.3: Worst MSE and MAE for the different models using Dataset 1. The best score for each metric is highlighted in bold. *#Samples* denotes the number of *training* samples.

Model	MSE $[(\text{EU})^2 \cdot 10^{-4}]$	MAE $[\text{EU} \cdot 10^{-3}]$	Product	#Samples
ANN	5.69	18.9	Product 5	55969
Linear regression	**4.65**	**17.28**	Product 5	55969
Gradient Boosted trees	7.51	22.2	Product 5	55969

4.3 Dataset 2

In this section statistics of the models trained on Dataset 2 are presented. The ANN model scored best for most metrics, compared to the other models, as can be seen in tables 4.4, 4.5 and 4.6. The metric for which the Gradient Boosted Trees model had the best score was for the product with best MSE. This was Product 8. The linear regression model scored best on MAE for the product with lowest MAE. This was Product 2.

Table 4.4: Total MSE and MAE for the different models using Dataset 2. The best score for each metric is highlighted in bold. *#Samples* denotes the number of *training* samples.

Model	MSE $[(\text{EU})^2 \cdot 10^{-4}]$	MAE $[\text{EU} \cdot 10^{-3}]$	#Samples
ANN	**1.96**	**9.47**	87292
Linear regression	4.02	13.50	87292
Gradient Boosted trees	2.28	10.3	87292

Table 4.5: Best MSE and MAE for the test data for the different models using Dataset 2. The best score for each metric is highlighted in bold. *#Samples* denotes the number of *training* samples.

Model	MSE $[(\text{EU})^2 \cdot 10^{-4}]$	MAE $[\text{EU} \cdot 10^{-3}]$	Product	#Samples
ANN	0.607	5.02	Product 1 (MSE) Product 12 (MAE)	87292
Linear regression	0.619	**4.77**	Product 2	87292
Gradient Boosted Trees	**0.508**	5.700	Product 8	87292

Table 4.6: Worst MSE and MAE for the different models using Dataset 2. The best score for each metric is highlighted in bold. *#Samples* denotes the number of *training* samples.

Model	MSE $[(\text{EU})^2 \cdot 10^{-4}]$	MAE $[\text{EU} \cdot 10^{-3}]$	Product	#Samples
ANN	**5.09**	**17.6**	Product 5	87292
Linear regression	8.64	23.5	Product 8	87292
Gradient Boosted Trees	6.84	21.3	Product 5	87292

Table 4.7 shows a summary of the MSE and MAE scores that are shown in the previous tables. The best score for each metric is highlighted in bold and shows that the ANN model has the higher score in most metrics and that the use of Dataset 2 results in better scores compared with Dataset 1.

Table 4.8 shows a summary of the MSE values calculated for the different data splits, training, validation, and test.

Table 4.7: Mean Squared Error (MSE) and Mean Absolute Error (MAE) for the three models on the two datasets (DS 1 and DS 2). What product had the best and worst MSE/MAE can be seen in tables 4.2, 4.3, 4.5 and 4.6. The best score for each metric is highlighted in bold.

Metrics	**ANN**			**GBT**			**Linear Regression**		
	DS 1	**DS 2**	**Diff**	**DS 1**	**DS 2**	**Diff**	**DS 1**	**DS 2**	**Diff**
Total MSE $[(\mathbf{EU})^2 \cdot 10^{-4}]$	2.25	**1.96**	-0.29	2.56	2.28	-0.28	2.04	4.02	1.98
Total MAE $[\mathbf{EU} \cdot 10^{-3}]$	10.7	**9.47**	-1.23	11.3	10.3	-1	9.79	13.50	3.71
Best MSE $[(\mathbf{EU})^2 \cdot 10^{-4}]$	0.609	0.607	-0.002	0.712	**0.508**	-0.204	0.685	0.609	-0.76
Best MAE $[\mathbf{EU} \cdot 10^{-3}]$	5.24	5.02	-0.22	5.66	5.70	0.04	5.26	**4.77**	-1.51
Worst MSE $[(\mathbf{EU})^2 \cdot 10^{-4}]$	5.69	5.09	-0.6	7.51	6.84	-0.67	**4.65**	8.64	3.99
Worst MAE $[\mathbf{EU} \cdot 10^{-3}]$	18.19	17.6	-0.59	22.2	21.3	-0.9	**17.28**	28.28	11.0

Model	Training MSE	Validation MSE	Test MSE
ANN	0.00019469	0.00019837	0.00019277
Linear Regression	0.00040473	0.00039991	0.00040248
Gradient Boosted Trees	0.00022423	0.00022306	0.00022776

Table 4.8: Training, validation, and test MSE for the different models trained on Dataset 2.

4.4 ANN

In this section, plots for the ANN model are shown. Figure 4.1 shows the normalized predicted energy consumption and the true target energy consumption data sorted for the target data. The figure shows that most energy consumption values lie below $\tilde{0}.45$ EU's (45 in the figure) but that there are values above this as well that the model is able to predict.

Figure 4.1: Plot of the predictions sorted by the target data (red dashed reference line) of Dataset 2. The ideal plot would have all the blue prediction points lying right on the dashed red line. The unit of the energy consumption on the y-axis is [EU·10^{-2}].

Figure 4.2 shows the predicted energy consumption against the target energy consumption with a reference line with slope one. Ideally, the points should all be on the reference line since this means that the predictions are equal to the target values. From Figure 4.3, which shows the distribution of the points in Figure 4.2, it can be seen that most of the points are concentrated on the reference line and that the points that lie on the furthest from the line are few in comparison to those closer to the line.

Figure 4.4 shows all the products in Dataset 2 and the energy consumption against the RB DL utilization for the NR cell.

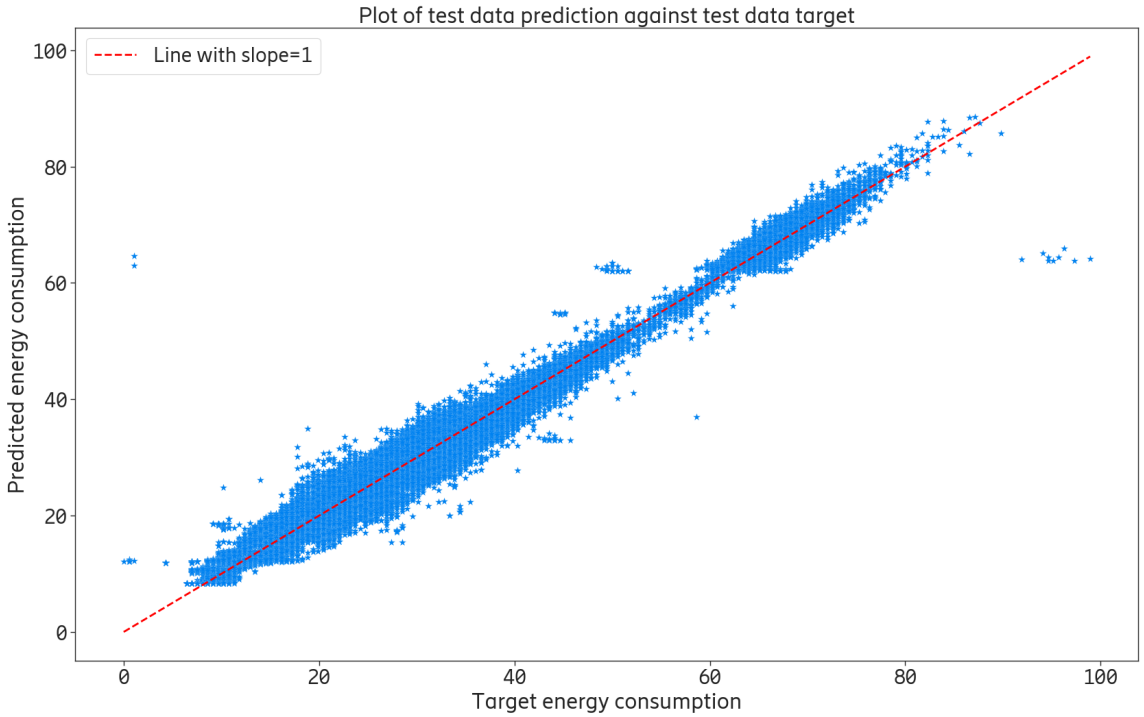


Figure 4.2: Prediction of test data plotted against target test data of the multi-day dataset. The ideal plot would have all the blue prediction points lying on the reference line with slope 1. A 3D distribution plot of the same data can be seen in Figure 4.3.

4.5 Gradient Boosted Trees

In this section plots for the Gradient Boosted Trees model are shown. Figures 4.5, 4.6 and 4.7 show the same thing as for the ANN model with a large part of the points in Figure 4.6 being concentrated on the reference line as shown in Figure 4.7. Dataset 2 is used for all plots.

4.6 Linear regression predictions

In this section plots for the Linear regression model are shown. Figure 4.8 shows predictions against target values for the product with the lowest MSE which is Product 2. Figure 4.9 shows the predictions against target values for the product with the highest MSE, which is Product 8. Dataset 2 is used for all plots.

4.7 Time series

In this section time series of the mean energy consumption over three days is shown for three products, Product 1 (Single band AIR product), Product 5 (Dual band Radio product), and Product 8

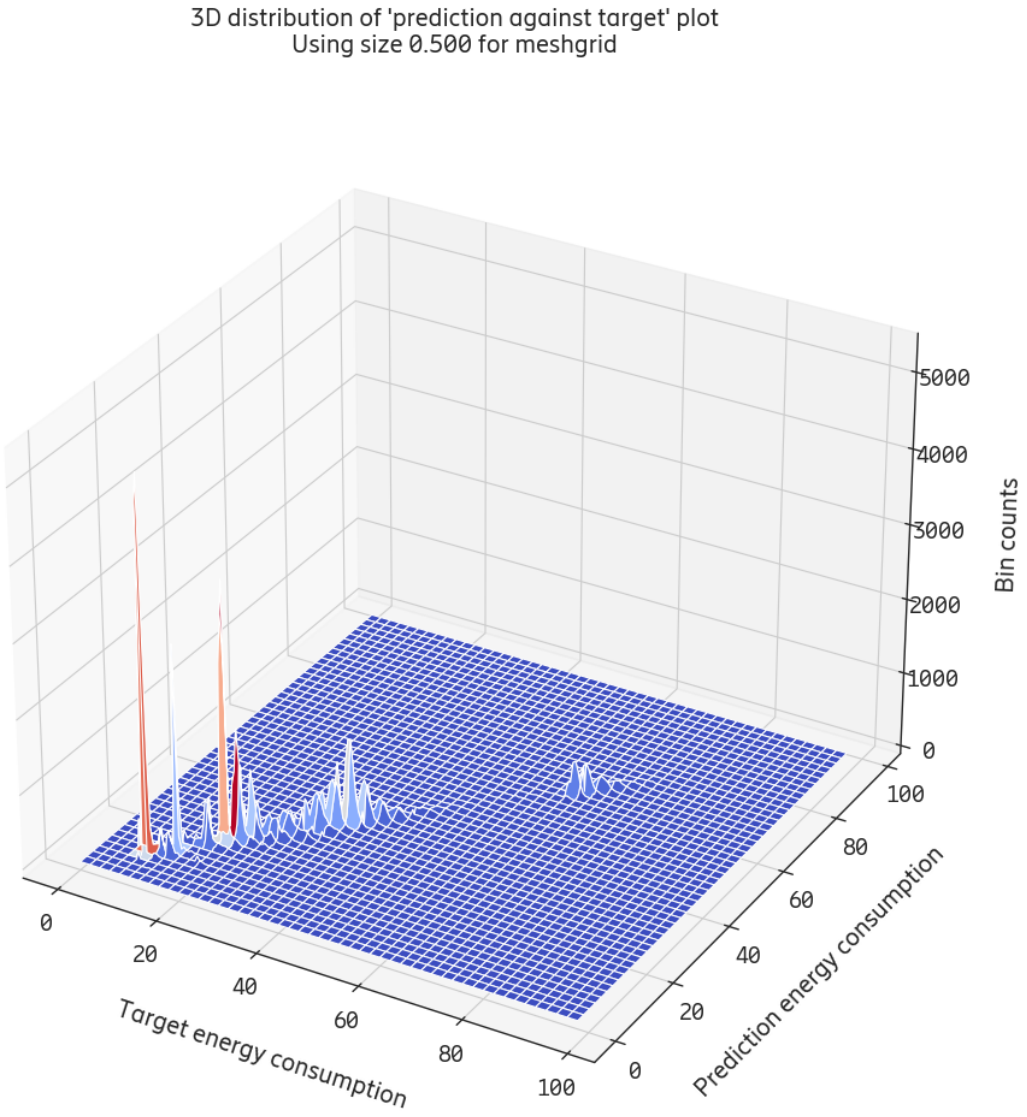


Figure 4.3: 3D distribution plot of Figure 4.2.

(Single band Radio product). The data used is from Dataset 2. The different time series are created by calculating the mean of the target energy consumption and the predicted energy consumption for each timestamp.

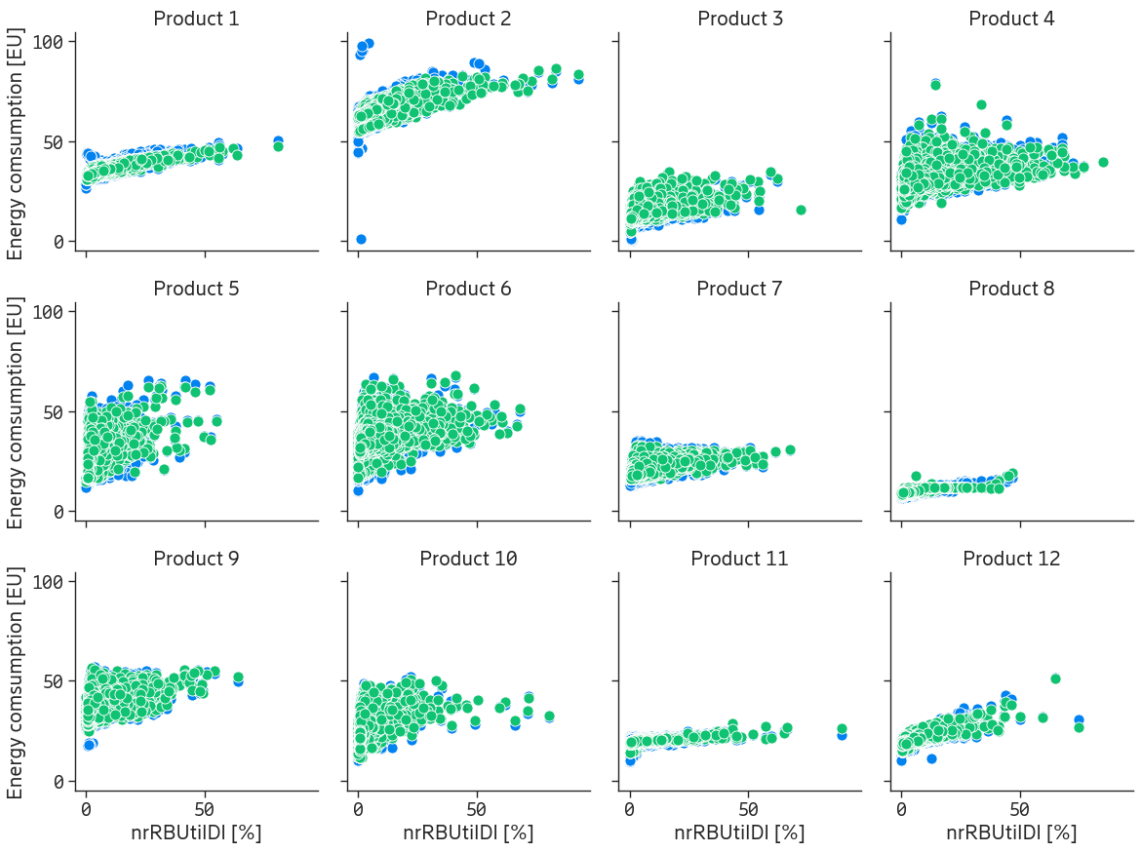


Figure 4.4: True energy consumption (blue) and predicted energy consumption (green) using the ANN model and Dataset 2 against RB symbol utilization (nrRBUtilDl) for every product.

4.7.1 Gradient Boosted Trees Time Series

Figures 4.10, 4.11 and 4.12 show the time series of the mean energy consumption over three days for Product 1, Product 5, and Product 8, respectively. The Gradient Boosted Trees model scored highest on MSE and MAE for Product 8 and lowest for Product 5 as seen in tables 4.5 and 4.6.

4.7.2 ANN Time Series

Figures 4.13, 4.14, and 4.15 show the same plots as for the Gradient Boosted Trees model but with predictions made with the ANN model.

4.7.3 Linear Regression Time Series

Figures 4.16, 4.17 and 4.18 show the time series as for the other two models but with predictions made with the linear regression model.

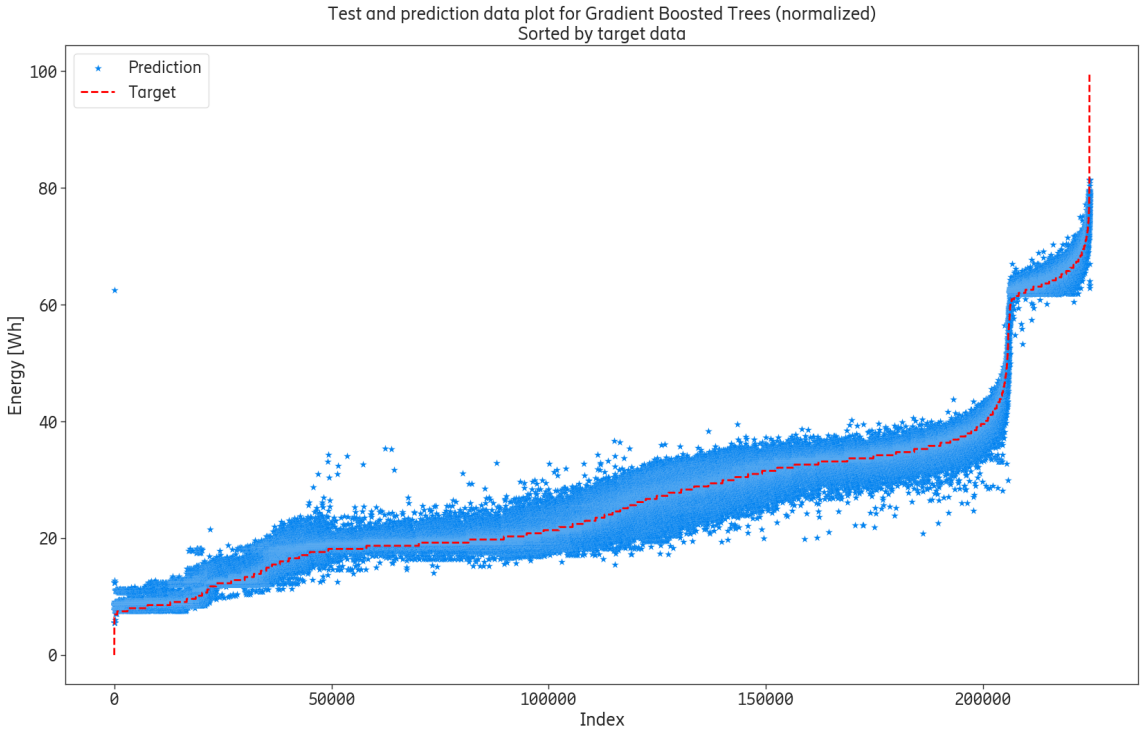


Figure 4.5: Plot of the predictions sorted by the target data (red dashed reference line) of the multi-day Dataset with the Gradient Boosted Trees model. The ideal plot would have all the blue prediction points lying right on the dashed red line.

4.7.4 Dual banded radios

Figures 4.19, 4.20, and 4.21 all show the time series plot of the mean prediction and target data for the same product. As can be noted there are some differences in how well the predicted mean is following the target mean, where the ANN model follows it much more closely than the linear regression model and gradient boosted trees model.

4.8 Inferences

In this section examples of inferences that can be done with the ANN model is shown.
Figure 4.22 shows simulated energy consumptions for Product 1 plotted against PRB DL utilization with three different configurations. The three configurations were combinations of downlink bandwidths of 100 and 40 Hz and max configured output power of 100 and 50 W. It also shows energy consumptions taken from field data for these configurations as well as simulated energy consumption generated in a laboratory. The laboratory generated data was made using an early version of Product 1 with the different configurations mentioned. The product was tested for 11 values of PRB DL utilizations in the interval $[0\%, 100\%]$. The figure shows that the laboratory

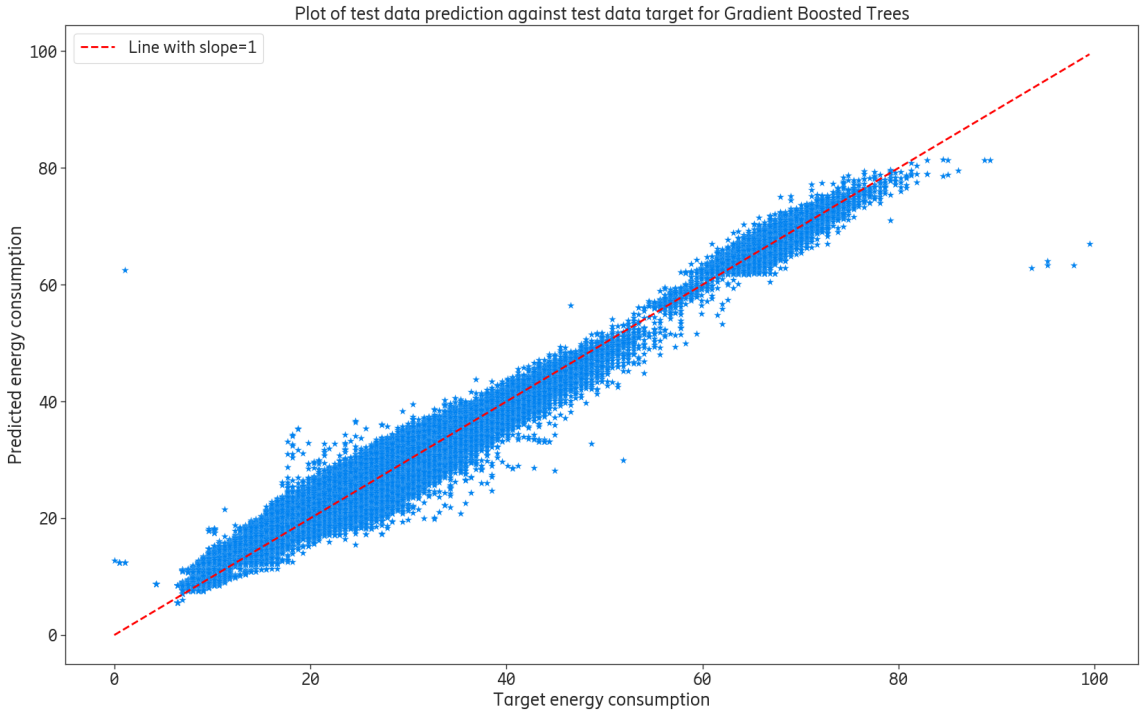


Figure 4.6: Prediction of test data plotted against target test data of Dataset 2. The ideal plot would have all the blue prediction points lying on the reference line with slope 1.

results differ from the simulated results, especially for higher PRB DL utilizations, where field data is more rare. Figure 4.23 is a smaller plot where only the PRB DL utilization interval where field data is present is shown.
Figure 4.24 shows what the ANN model outputs when changing an AAS/AIR-product with 32 antennas to instead have 64 antennas and vice versa. Product 1 uses 32 antennas originally and product 2 uses 64 antennas originally.

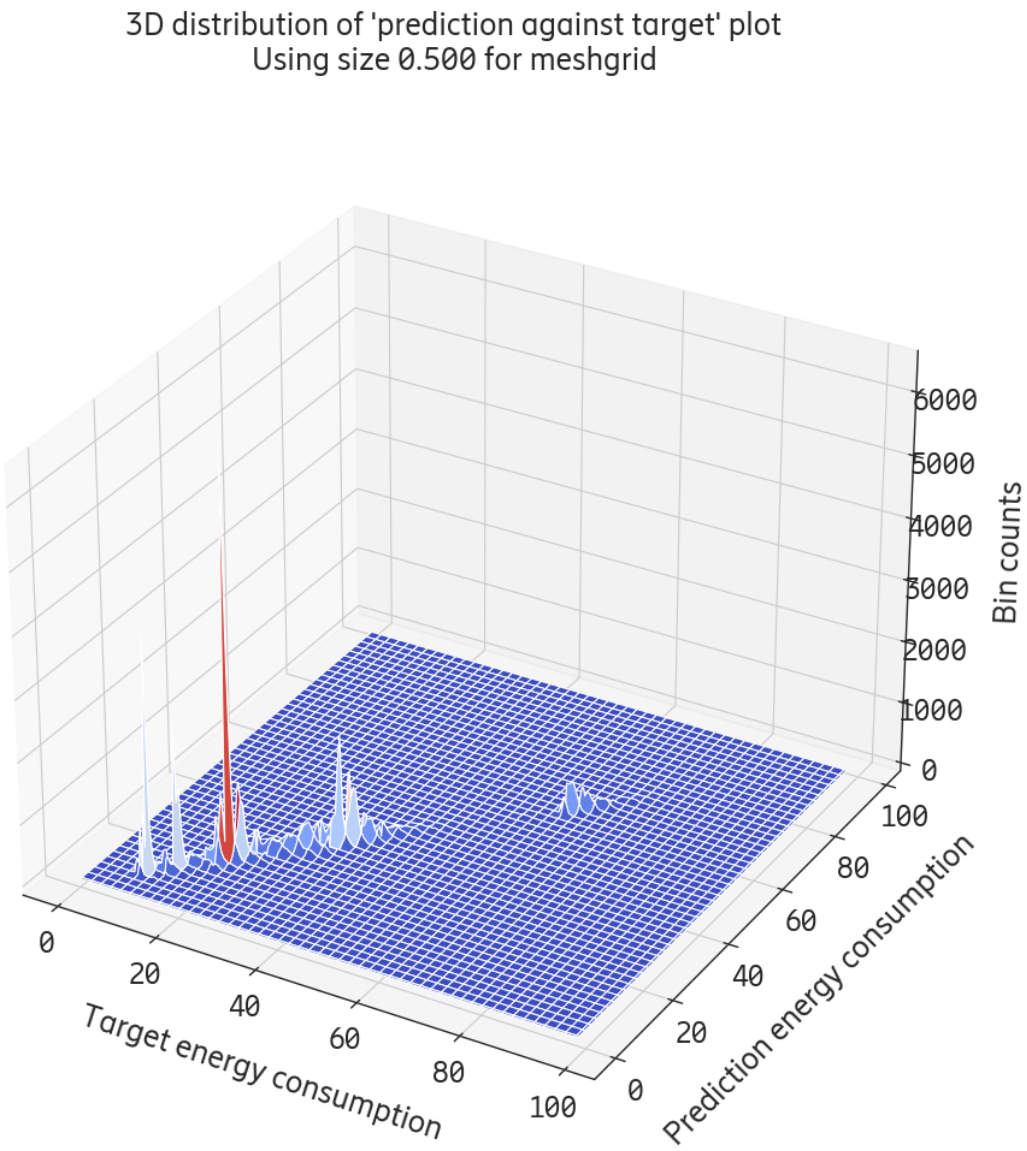


Figure 4.7: 3D distribution plot of Figure 4.6.

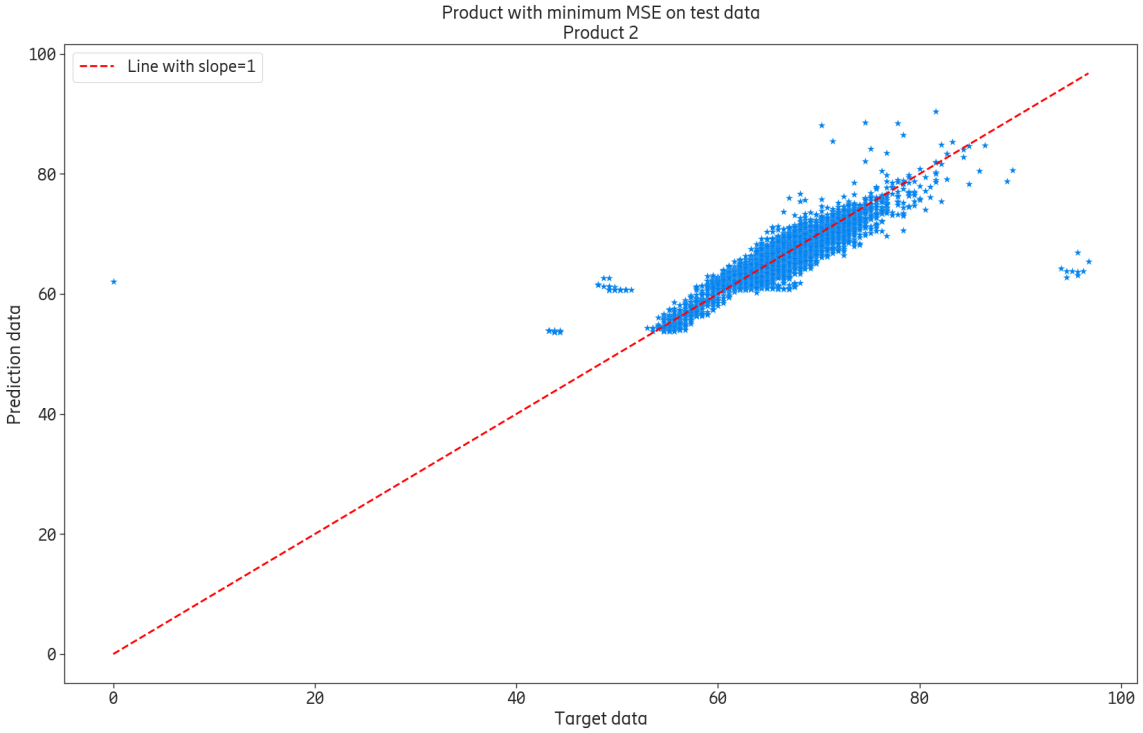


Figure 4.8: Prediction against target plot for a product for which linear regression model performed best, in the meaning of having the lowest MSE.

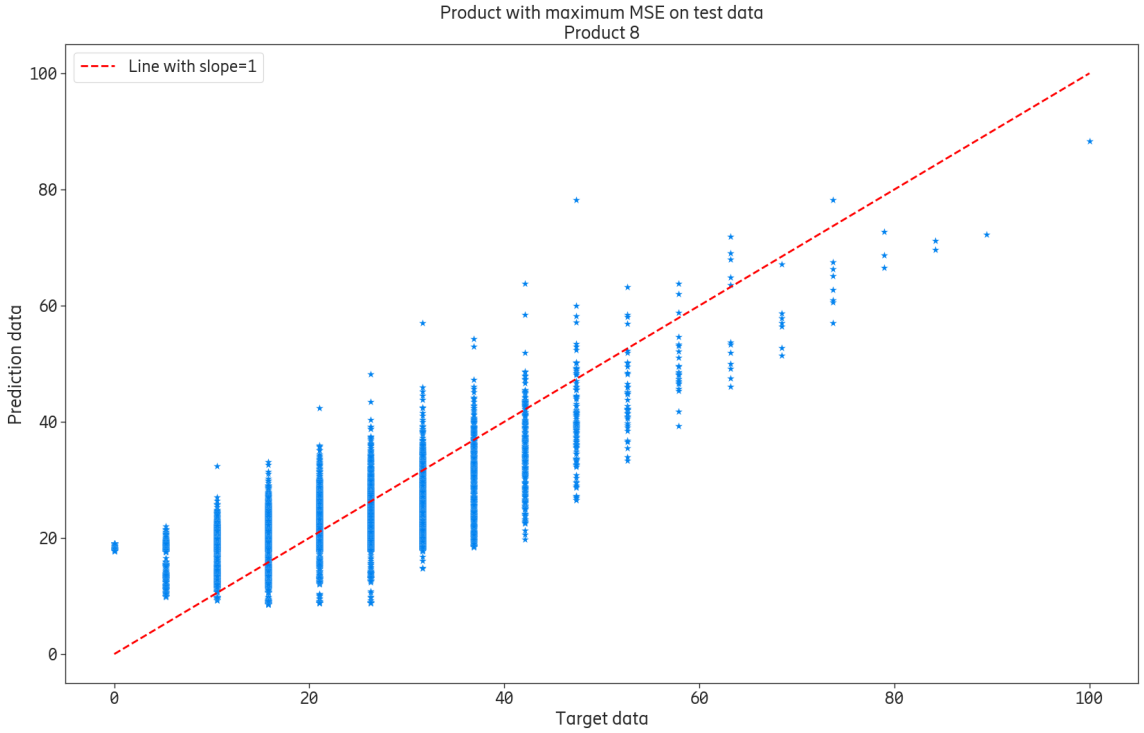


Figure 4.9: Prediction against target plot for a product for which linear regression model performed worst, in the meaning of having the highest MSE.

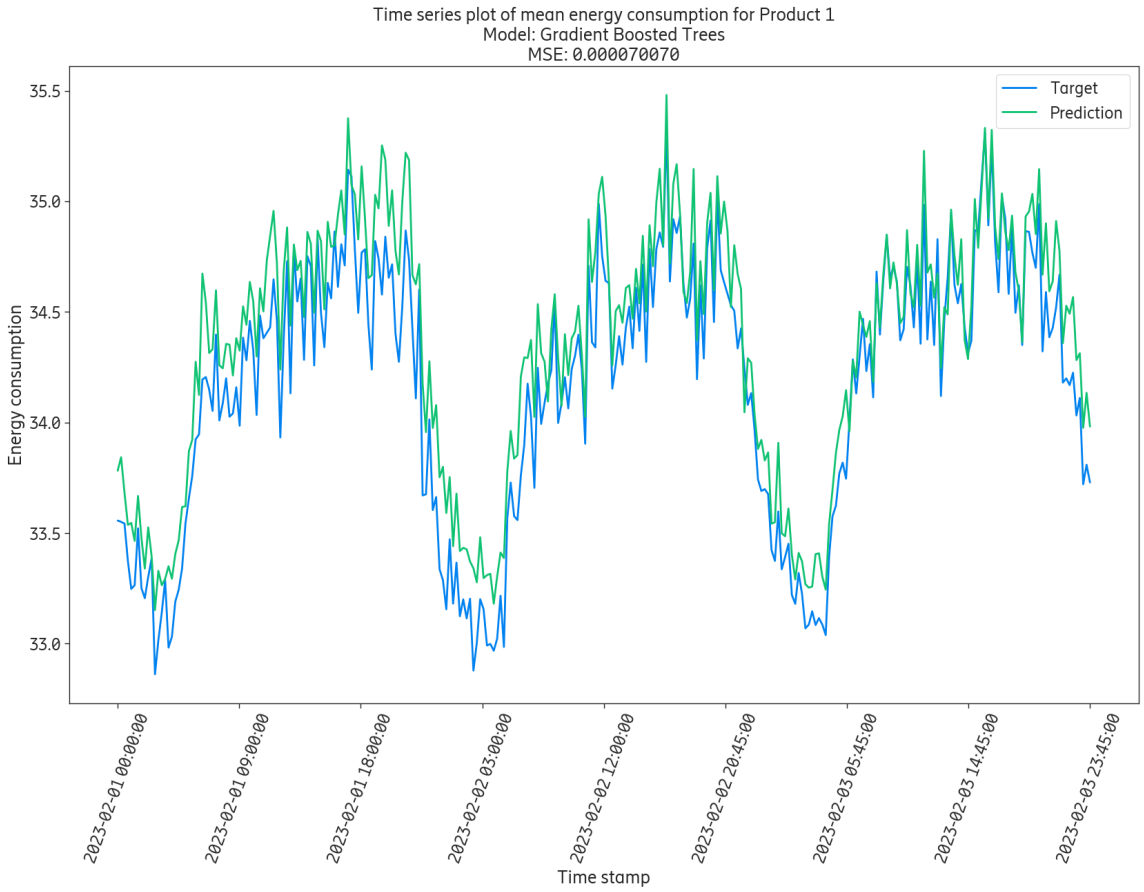


Figure 4.10: Mean energy consumption over three days for Product 1 with true target energy consumption in blue and predicted mean energy consumption using the Gradient Boosted Trees model in green. The MSE for Product 1 was $0.701 \cdot 10^{-4}$.

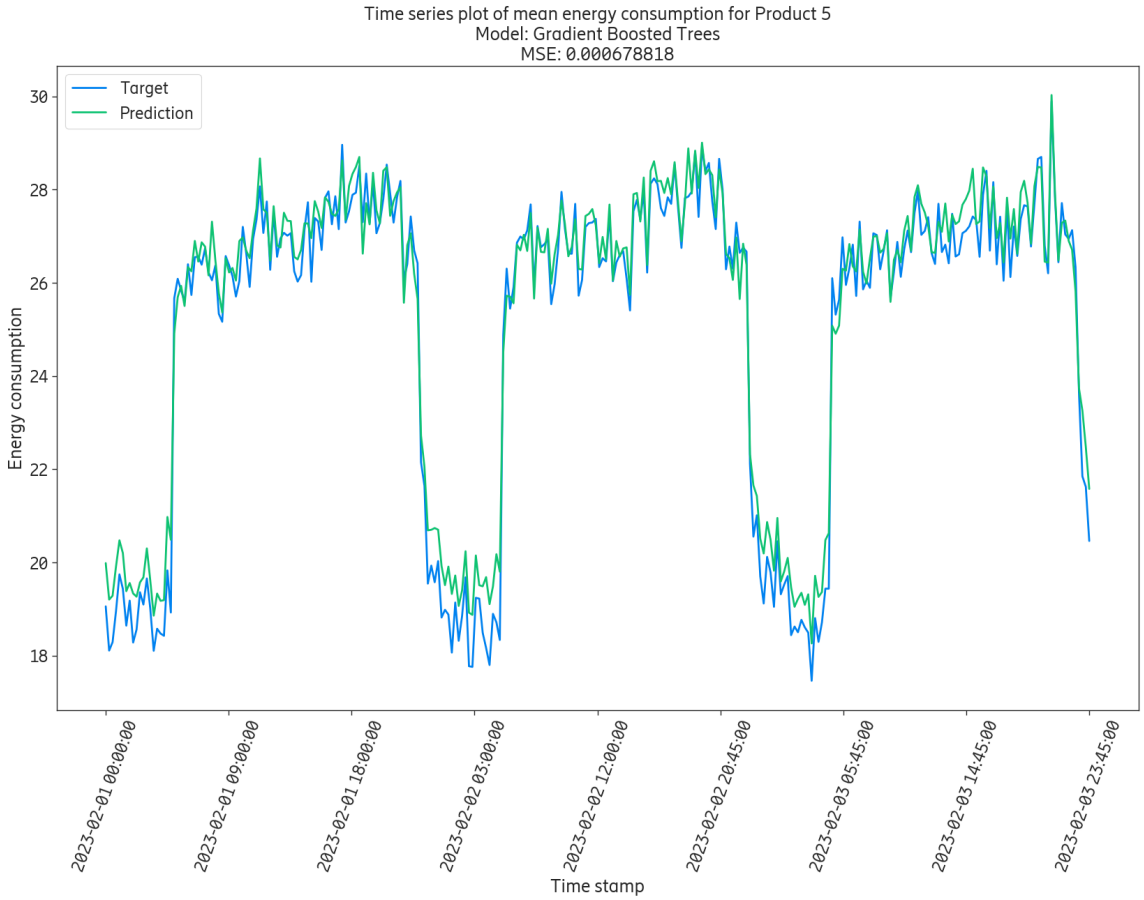


Figure 4.11: Mean energy consumption over three days for Product 5 with true target energy consumption in blue and predicted mean energy consumption using the Gradient Boosted Trees model in green. The MSE for Product 5 was $0.679 \cdot 10^{-4}$.

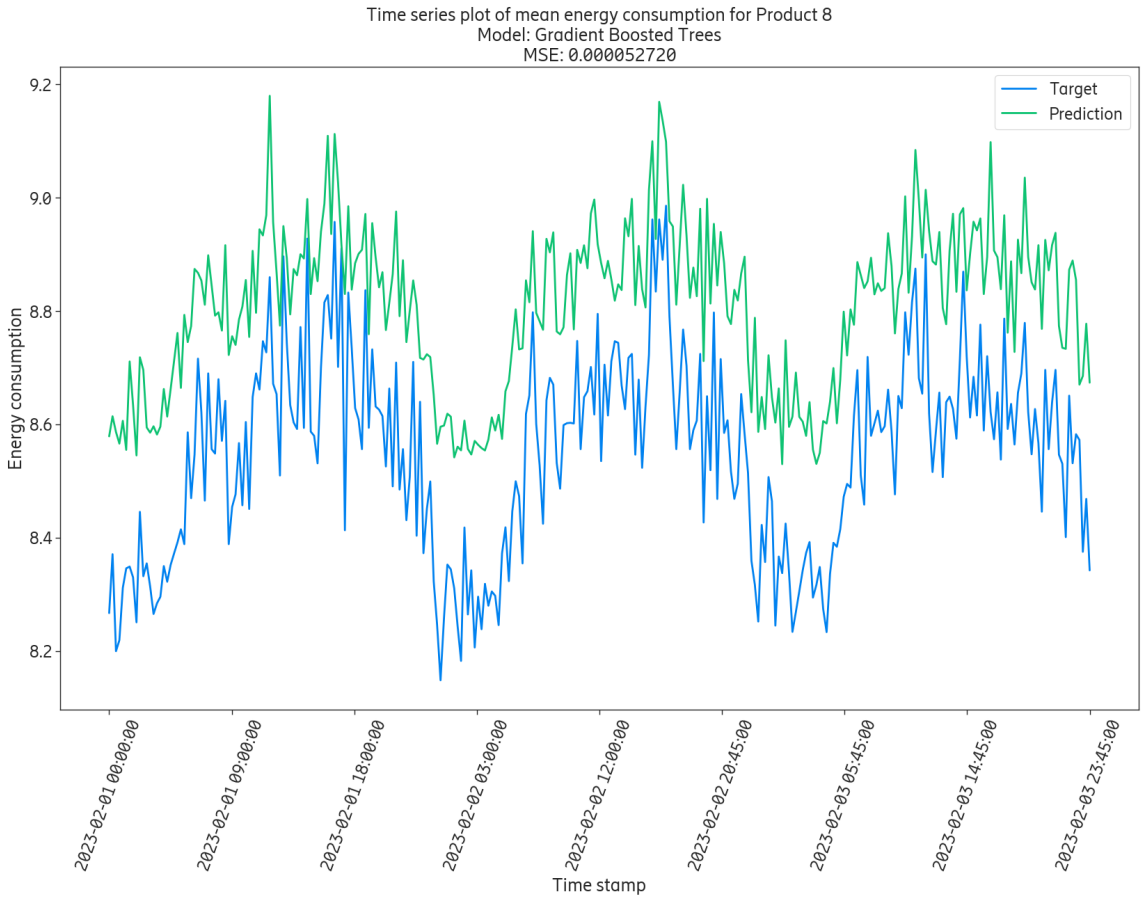


Figure 4.12: Mean energy consumption over three days for Product 8 with true target energy consumption in blue and predicted mean energy consumption using the Gradient Boosted Trees model in green. The MSE for Product 8 was $0.527 \cdot 10^{-4}$.

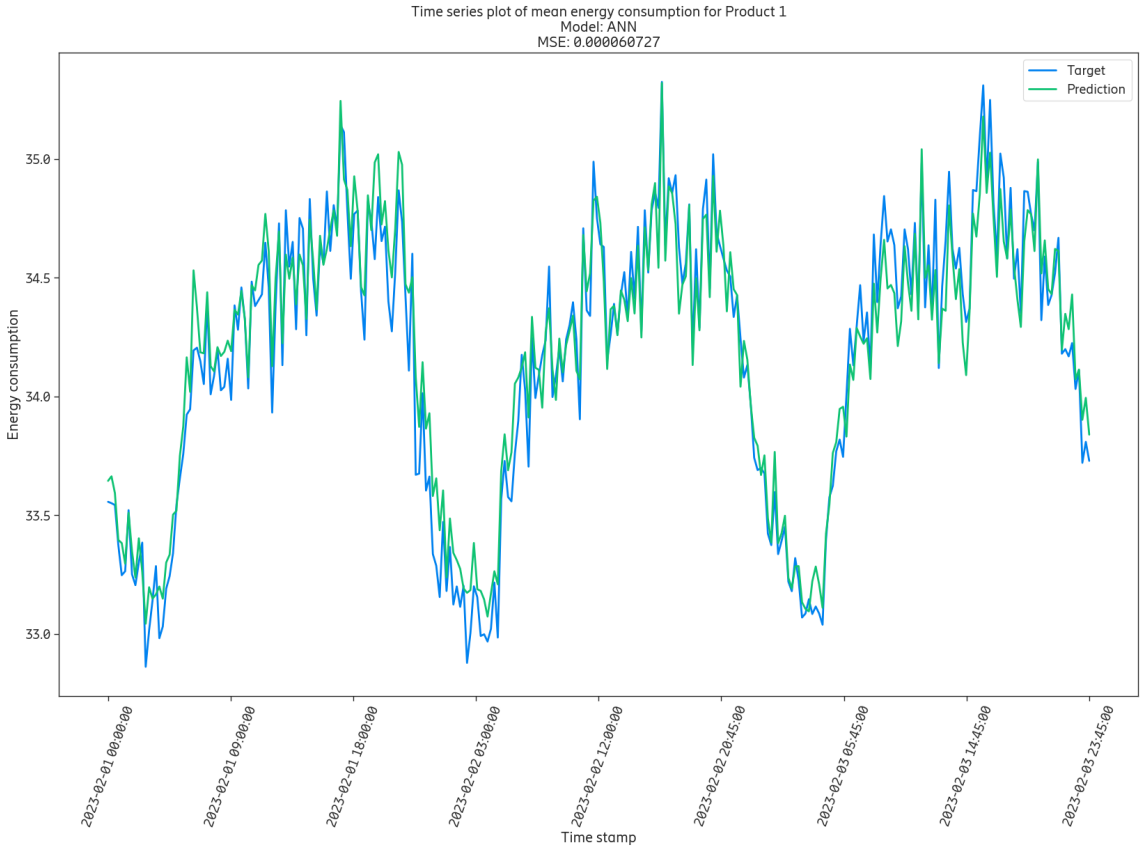


Figure 4.13: Mean energy consumption over three days for Product 1 with true target energy consumption in blue and predicted mean energy consumption using the ANN model in green. The MSE for Product 1 was $0.607 \cdot 10^{-4}$.

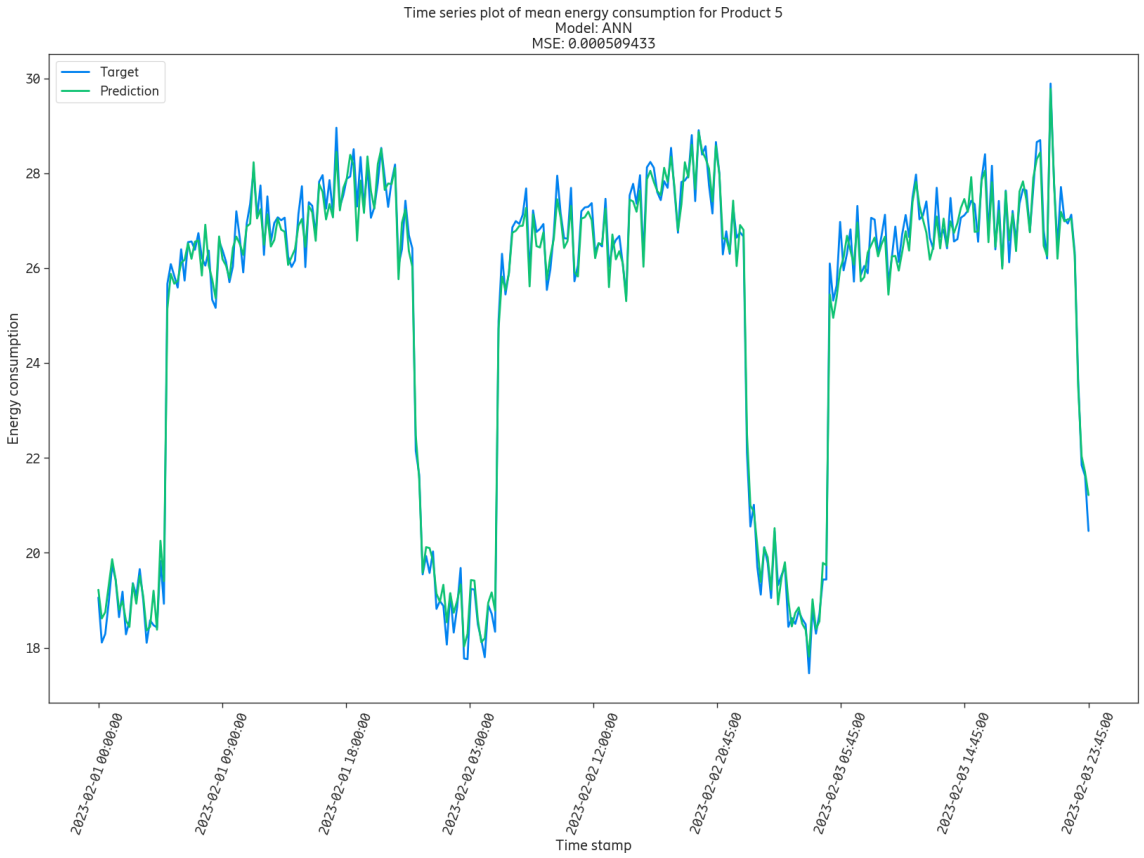


Figure 4.14: Mean energy consumption over three days for Product 5 with true target energy consumption in blue and predicted mean energy consumption using the ANN model in green. The MSE for Product 5 was $5.09 \cdot 10^{-4}$.

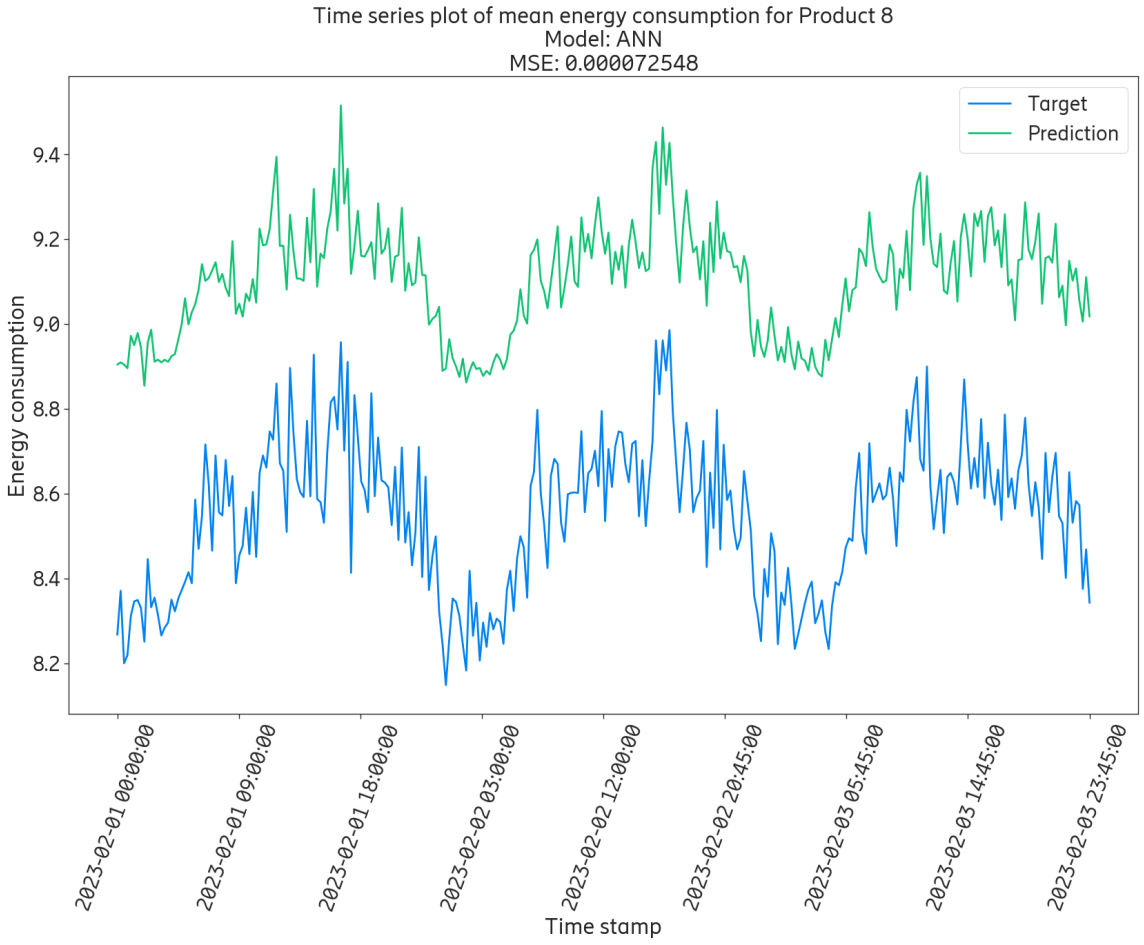


Figure 4.15: Mean energy consumption over three days for Product 8 with true target energy consumption in blue and predicted mean energy consumption using the ANN model in green. The MSE for Product 8 was $0.725 \cdot 10^{-4}$.

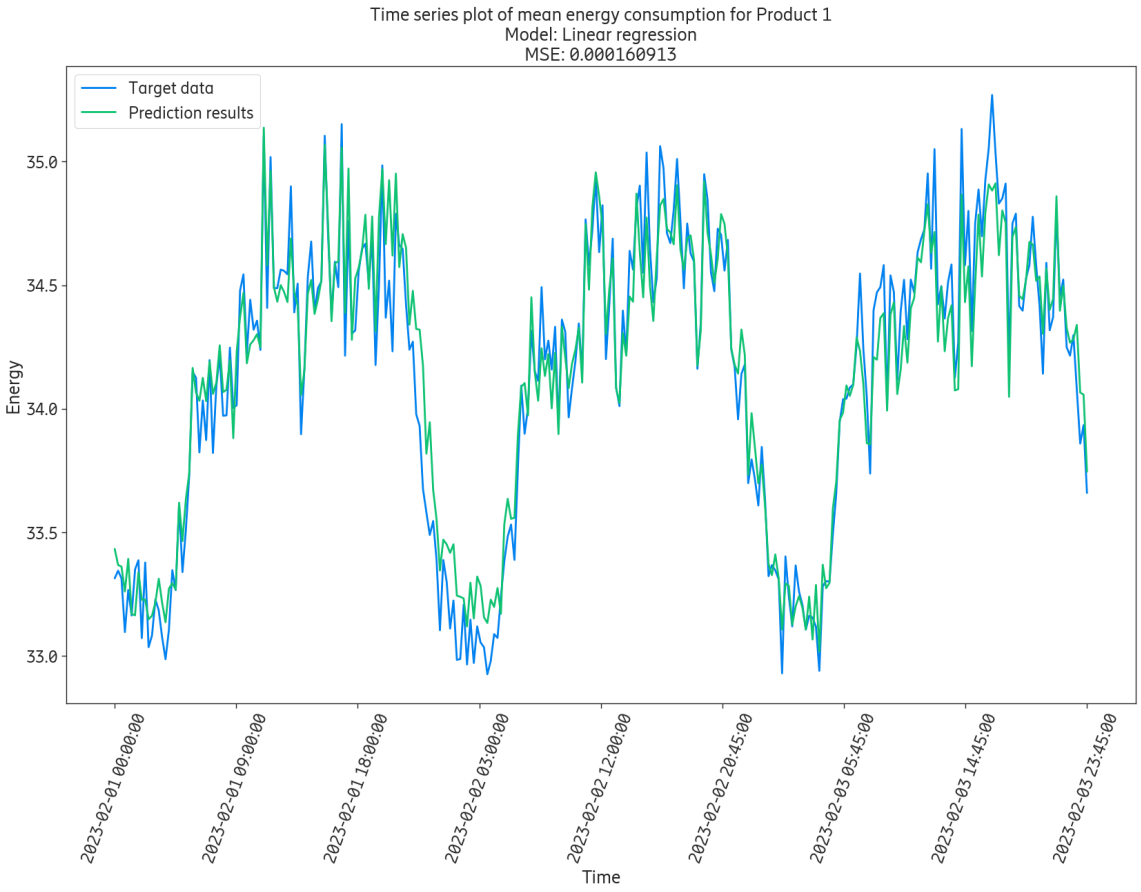


Figure 4.16: Mean energy consumption over three days for Product 1 with true target energy consumption in blue and predicted mean energy consumption using the Linear regression model in green. The MSE for Product 1 was $1.61 \cdot 10^{-4}$.

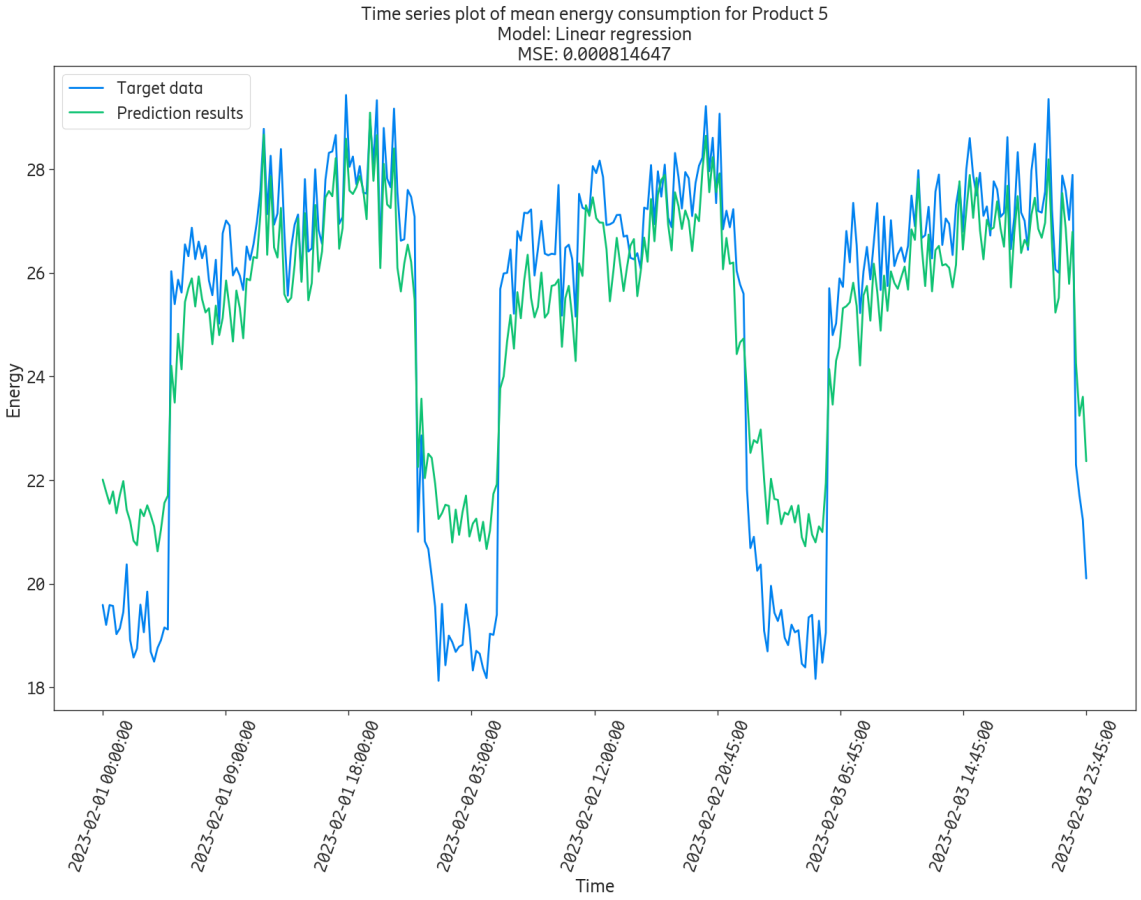


Figure 4.17: Mean energy consumption over three days for Product 5 with true target energy consumption in blue and predicted mean energy consumption using the Linear regression model in green. The MSE for Product 5 was $8.15 \cdot 10^{-4}$.

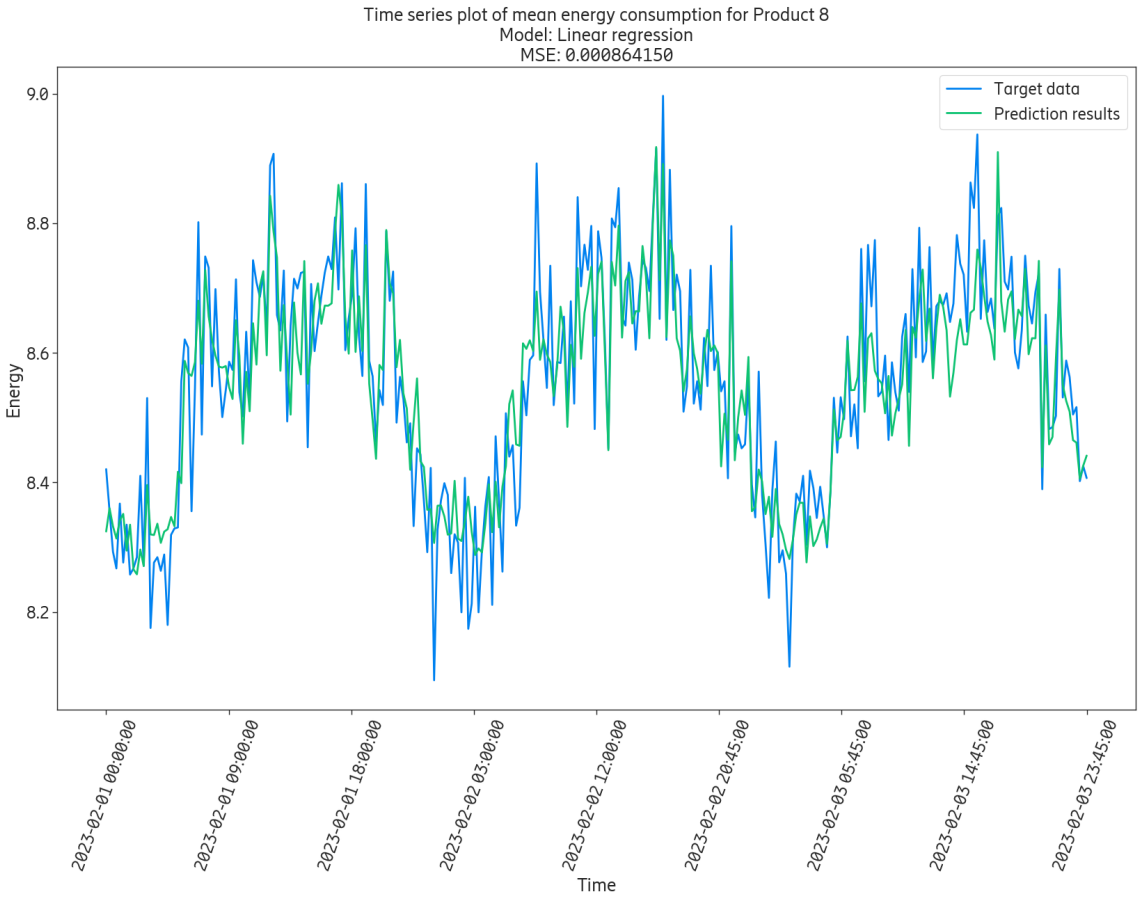


Figure 4.18: Mean energy consumption over three days for Product 8 with true target energy consumption in blue and predicted mean energy consumption using the Linear regression model in green. The MSE for Product 8 was $8.64 \cdot 10^{-4}$.

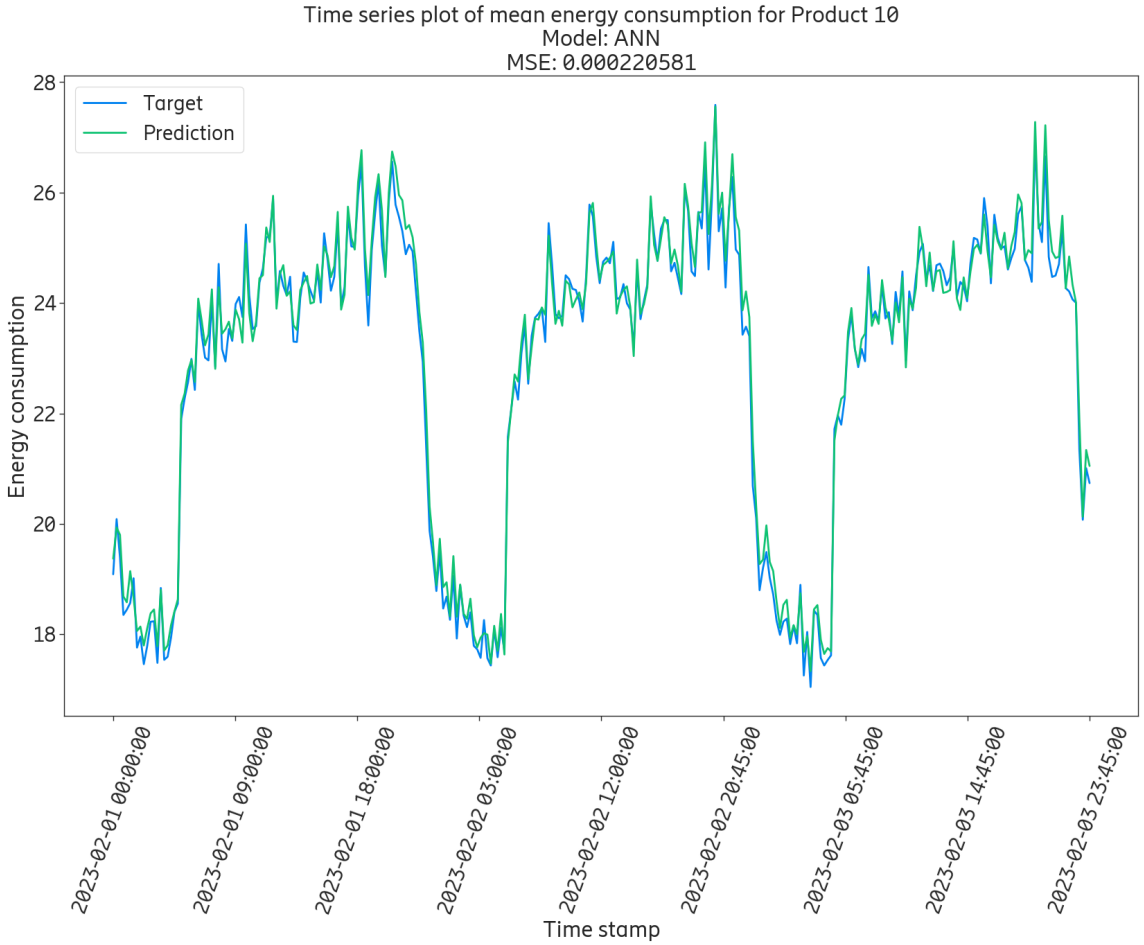


Figure 4.19: Time series plot of a dual-band product with ANN model used for prediction of Dataset 2.

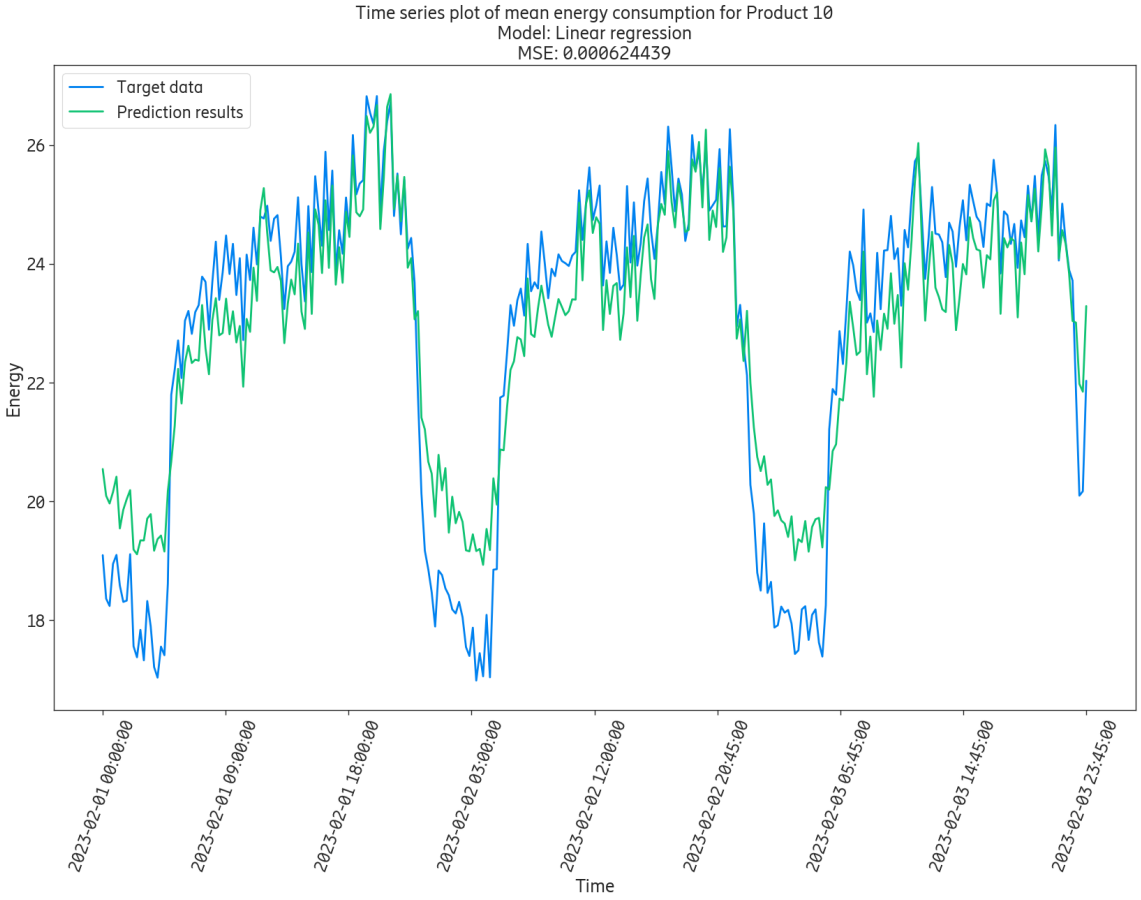


Figure 4.20: Time series plot of a dual-band product with linear regression model used for prediction of Dataset 2.

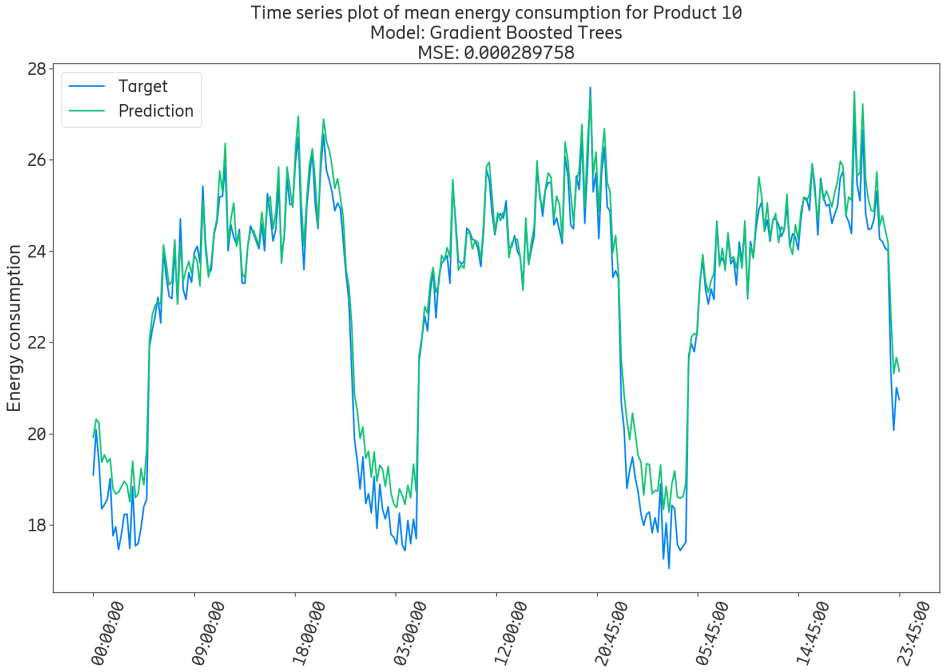


Figure 4.21: Time series plot of a dual-band product with gradient boosted trees model used for prediction of Dataset 2.

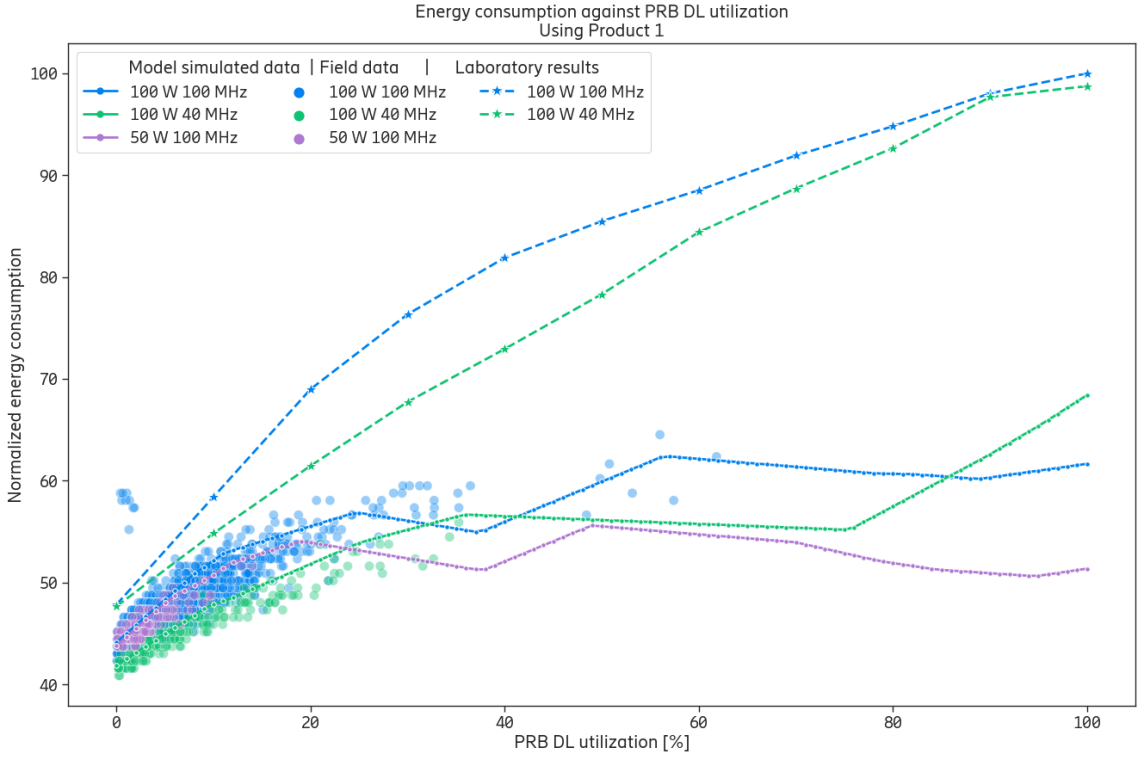


Figure 4.22: Plot of energy consumption against PRB DL utilization. Three categories are present, results when simulating using the ANN model (line with dots), observed field data (scatter plot with circles), and lab results (dashed line with stars).

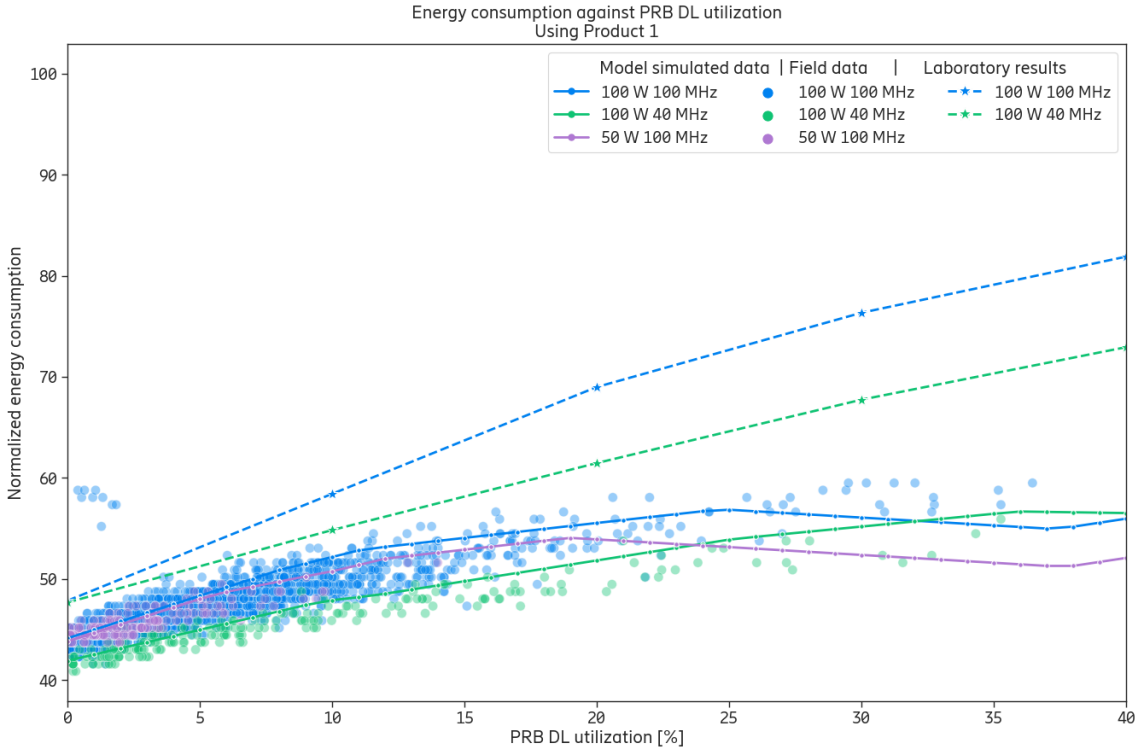


Figure 4.23: Plot of energy consumption against PRB DL utilization. Three categories are present, results when simulating using the ANN model (line with dots), observed field data (scatter plot with circles), and lab results (dashed line with stars). Zoom in of Figure 4.22

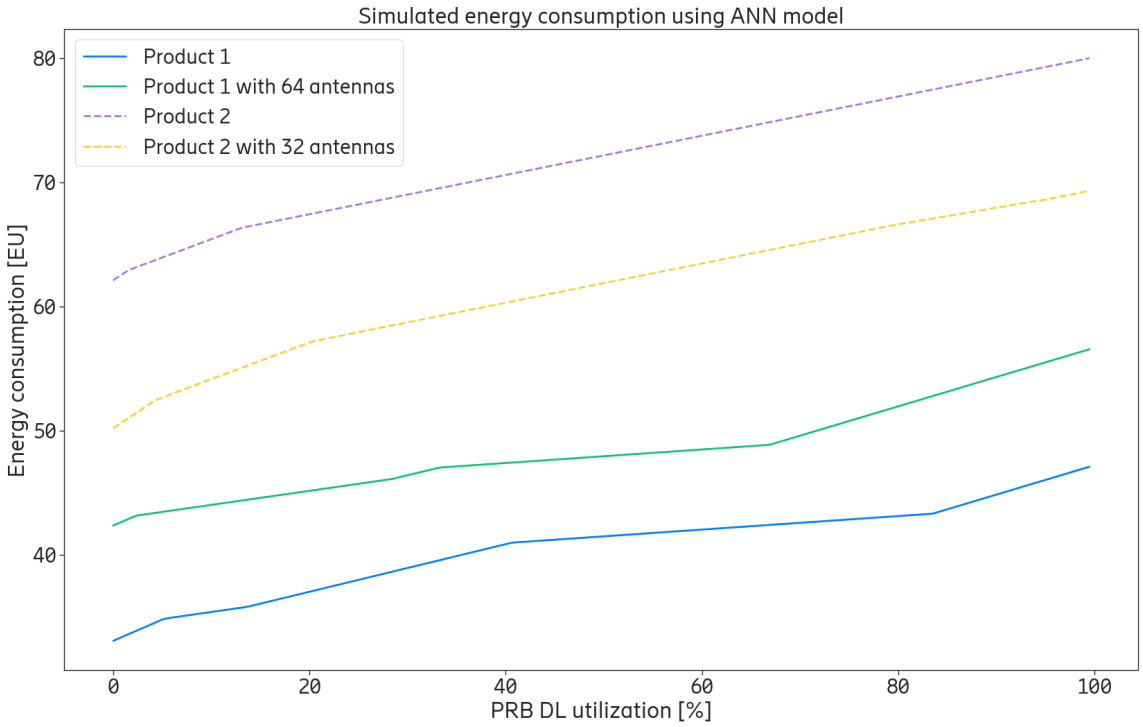


Figure 4.24: Model output when changing from 32 to 64 TX/RX antennas and vice versa on AAS products.

Chapter 5

Discussion

5.1 Dataset

Studying field data is usually very interesting but is also subject to constraints. Aspects such as what kind of products are present, what kind of use cases the products are subject to, data distribution, etc. The implication of this is that the models are not especially good at extrapolating to samples it has not observed. Usually, this is not a problem since if the goal is to model field data, then these edge cases that are not found in the field data will never exist, thus removing the need for it. The range of data available is dynamic since it depends on the amount of radio base stations that are installed in the field. Because of this, time will, probably, give a bigger range of available values.

It can be seen in Figure 4.22 how the model fails at high PRB DL loads since there are a non-existent number of samples in that region. It is fair to assume that this is not just the case for this feature, but that many features are not evenly distributed between their highest and lowest values. With Dataset 2, where there were equally many samples for each product, the model performed better than with Dataset 1. With more evenly distributed data across more features, the model performance may be improved.

In this work, there were some limitations on the feasible amount of data used to train the models. The data collection part is not straightforward since the data has to be pulled from widely different places and mapped together. This creates constraints on the available hardware, the capacity of the data collection tools, and man hours to set up the data collection pipelines. The actual amount of data that is available is in the magnitude of petabytes and it is thus not the constraining factor, instead, it is how that much can be gathered and utilized efficiently that is constraining.

The choice of a single customer might seem illogical since from the outside this would imply that the models are only adjusted for one single customer. However, the customer of choice has the advantage for one of the features used, *configured max TX power*, that it is more distributed over possible values. A lot of other customers have it set to the maximal value, thus not giving the model a chance to see different configurations for this setting. However, the choice of only one customer can have implications on the generalizability of the model, since over-fitting on this one customer may occur. To properly draw a conclusion weather the model is accurate on data from

other customers, further tests are needed.

5.2 Energy Consumption predictions

The model and dataset that gave the best scores in terms of total MSE and total MAE was the ANN model using Dataset 2, see Table 4.7. For the ANN model and the Gradient Boosted Trees model it is expected that a more balanced dataset gives a better result. Both models rely on looking at observations and being "guided" toward an optimal solution by the training. If the dataset is skewed, where there are many more observations of one product, the model can be skewed to try to correctly estimate for that product, since the overall loss would be less. When balancing the dataset the model will not suffer from this and thus Dataset 2 unlocks more of the advantages of more complex models than linear regression.

With the best total MSE and MAE for Dataset 2, the ANN model seems to be the better model in terms of being more generalized. On Dataset 2 the Gradient Boosted Trees model had the lowest MSE for its product with the best predictions and the linear regression model had the lowest MAE for its product with the best predictions. On Dataset 2 the ANN model performed the worst on data from Product 5 compared to other products but still achieves a better score than the other models for the same product, see Table 4.6. Table 4.7 shows that the linear regression model had the lowest score for the predictions on the worst model using Dataset 1. With more data in Dataset 2, this is no longer the case and the reason may be because of the valleys that can be seen in Figure 4.17. As previously mentioned, Dataset 1 is made up of data from only one day which limits the number of low energy consumption measurements that occur during nighttime hours, where the linear regression model gives rise to the largest errors. By looking at the time series plots for this product for the different models (figures 4.14, 4.11 and 4.17) it seems that the ANN model better handles larger ranges of energy consumption values, with values ranging from approximately 0.18 to 0.29 EU's. For Product 1 the time series seen in figures 4.13, 4.10 and 4.16 look fairly similar but the MSE seen in the figures show that the ANN model and the Gradient Boosted Trees model outperforms the linear regression model with MSE scores approximately 2.5 times lower. For product 8 the mean energy consumption ranges from approximately 0.082 to 0.089 EU's and the time series seen in figures 4.15, 4.12 and 4.18, shows that the predictions made using the ANN model and the Gradient Boosted Trees model are offset with the predictions being systematically higher than the true energy consumption, suggesting a bias in the predictions. The linear regression model gives better predictions for the mean energy consumption for Product 1 compared to the other models, as seen in the figures. The MSE for the linear regression model is however larger than that of the other models which suggests that the linear regression model makes predictions with larger errors than the other models but has residuals with mean closer to zero.

Looking at Table 4.7, one can be confounded when looking at how the MSE and MAE changes between Dataset 1 and Dataset 2 for linear regression. Overall, the models perform worse for Dataset 2. The reason for this is that when going from Dataset 1 to Dataset 2, there is a radical change in the number of samples for each product. Observing Table 4.2, linear regression gets the best MSE for product 4 which has over 250,000 samples at its disposal compared to the meager sum of about 87,000 samples used per product in Dataset 2. The number of samples does not explain everything since in, again, Table 4.7, the best MAE is achieved by a product with around 79,000 samples.

Another interesting thing to notice is that when comparing with the results for product 2. Looking at Table 4.5, product 2 has become the best performing product using Dataset 2 with fewer samples, indicating that linear regression is not complex enough, even when using higher order polynomial terms, to predict energy for this product. Product 2 uses around 159,000 samples in Dataset 1 and the normal amount in Dataset 2. This gives further arguments why it is of interest to investigate more general models, such as ANN, that can both predict products with greater accuracy, while also being more generalized.

5.3 Training

During the training of the ANN model, no regularization methods ended up being used since no obvious signs of overfitting was detected. Table 4.8 shows that the loss (MSE) was very similar for the training, validation and test set. This suggest little overfitting and a balanced split of the dataset into training, validation and test datasets. The same reasoning is applied to the Gradient Boosted Trees model. The scores for the linear regression model also suggest a balanced split, with fairly similar MSE scores, although they differ more than for the other models.

The architectures and hyperparameter choices for the machine learning models were chosen as those that gave the best score on the validation set. It is fair to say that trying different architectures and further tuning the hyperparameters can result in a better model, as well as more and better data as described in Section 5.1.

5.4 Inferences by the ANN model

Figure 4.24 show the energy consumption for two AAS/AIR products plotted against PRB DL utilization. The plot suggests that Product 1 is more energy efficient than Product 2. It is worth noting that this does not take into account the effect of changing the number of antennas on other performance measurements, for example, throughput volume. Also worth noting is that the design and architecture of the hardware (public address system etc.) may differ between the "base" products, i.e., even if only 32 antennas are modelled for Product 2, it may suffer from having oversized hardware from the beginning.

Figures 4.22 and 4.23 show a simulation test for some different configurations of the features *configured max TX power* and *downlink bandwidth*. The values of the model simulations are centered around the field data as long as there are field data for that value. For the parts where field data is scarce, the model has a hard time getting a reasonable answer. This could be improved with a wider range of values for the field data but is also, as mentioned before, one of the constraints. This shows a weakness of the model since it seemingly does not understand that an increase in utilization should also result in an increase in energy consumption.

It should also be noted that the laboratory results in Figure 4.22 and Figure 4.23 was done on an older model version of the product in a laboratory setting which can be a reason that it shows higher energy consumption. Products have been shown to consume somewhat less energy in the field, of which one reason is the ambient temperature, cooling the product.

5.5 Single-band and multi-band products

In the datasets, products that operate multiple bands exist. There are single, dual, and triple-band products. When observing the predictions made by the ANN model and the linear regression model of the dual-band Product 5 in figures 4.14 and 4.17, and comparing them to the predictions made for the dual-band Product 10 in figures 4.19, 4.21 and 4.20 it is obvious that the linear regression model and the gradient boosted trees model have a harder time adjusting to the quicker oscillations. This could be a result of a simplification in the linear model. Linear regression has the inherent property that it will linearly combine data volumes, utilization, etc. from the different network bands. Some complexity can be added, as was done in this thesis, where polynomials of the features are added as independent features but the function is still constrained to a polynomial. This might not be a complex enough representation of what is actually happening inside the product. The ANN model takes care of this function complexity with the downside that it is not as easy to understand as a linear regression model.

When comparing with single band products, such as Product 1 with linear regression in Figure 4.16 with Product 1 with the ANN model in Figure 4.13 there does not seem to be that much of a dramatic difference. This indicates that, for the linear regression model, there is not much of an advantage to using a more complex model for the single-band products. These results suggest that for more complex products, the ANN model is preferable and that for more simple products, the linear regression approach could be used with relatively good performance.

5.6 Future work

To better predict energy consumption, suggested work can be done to improve the model performances:

- Larger amount of data used in model training
- Probe for data that represents a wider interval of possible feature values
- Include more complex radio base station configurations
- Include more energy-saving features
- Run the data collection when more radios and base stations have been installed
- Test different ML methods, for example, Random Forest and Weighted Average Ensemble Method
- How ML can be used for energy consumption optimization and energy conservation strategy selection

The complexity of the radio base stations in this work was limited to sites with one Field Replaceable Unit, one energy meter, and at least one NR cell. This is reasonable as a start since the lower complexity makes modelling easier and considering the fact that the simpler configurations are very common, the limitations are not too damaging in terms of the model's usefulness. More complex configurations, such as sites with two Field Replaceable Units are however also common and it

would be interesting to consider these sites when further improving on our models.

Energy-saving features are obviously important to include when modelling energy consumption. One such feature that was included was MicroSleepTime. This performance counter exists for both LTE cells and NR cells, but the counter was only available in the RDI for LTE cells. The performance counter for NR was implemented during the time of this thesis and therefore made it hard to find field data for the parameter, even though the energy-saving feature exists in the field. Features such as MIMO-sleep would also be interesting to include in the models.

As more radios and base stations are installed for 5G, the models can be improved by re-running the data collection and training the same models on this new data. This is believed to improve performance since it enables more data, both in terms of bulk and in terms of newly deployed features and measurements such as MicroSleepTime for NR, etc.

There are many machine learning methods that look promising for modeling energy consumption data. From [22], it seems that Random Forest could be of interest. The dataset in [22] consisted of data from residential buildings which is different than the dataset containing data from radio base stations. However, Random Forest has similarities with Gradient Boosted Trees, which worked well for modelling radio base station data. This indicates that Random Forest is of interest in further studies. In [16] the results found that an ensemble method, Weighted Average Ensemble Method, which combines Gradient Boosted Trees with Random Forest, outperformed both Gradient Boosted Trees and Random Forest used individually. Due to time constraints, only Gradient Boosted Trees were tested, but both Random Forest and Weighted Average Ensemble Method are of interest in further studies.

Furthermore, [12] achieves a loop where they use historical data to predict energy consumption. Then as the device is running, real-time values are passed back into the model to further improve the neural network. Achieving such a loop would improve prediction which would then lead to the opportunity to let a machine learning algorithm decide an energy conservation strategy. A different approach, done by [15], is to derive an analytical function with the help of a neural network which is used for optimization. Based on the results presented in the aforementioned papers, the potential for promising advancements in optimization and strategy selection using the ANN is apparent.

Chapter 6

Conclusion

The goal was to investigate different machine learning models predictive ability on energy consumption for radio base stations and if a general model that can handle multiple products could be found. We found that both an ANN and Gradient Boosted Trees achieves this goal with relatively good performances compared to a more naive approach where linear regression models were fitted for each product separately. The MSE for the best ANN and Gradient Boosted Trees models were in terms of energy units (EUs) squared $1.96 \cdot 10^{-4}$ and $2.28 \cdot 10^{-4}$ respectively. The naive model scored $4.02 \cdot 10^{-4}$. Furthermore the two former models seemed to capture data for the more complex dual band radio products better than the naive approach.

What could be observed in the results was that the ANN and the Gradient Boosted Trees model performed better for products where field data that had a larger spread was present, in the sense that there is a large difference between the maximum and the minimum of the observed values for energy. When comparing the time series plots, the ANN model seems to better approximate the behaviour of more complex products on a network level, where many units are present. Compare Figure 4.19 with Figure 4.20.

The features that resulted in the best performance were, *Number of antennas*, *Configured Max Transmitter Power*, *Frequency*, *Bandwidth*, *IoT capability*, *Product type*, *MAC volume*, *Throughput volume*, *PRB utilization*, *RB symbol utilization* and *MicroSleepTime*. These features resulted in 85 inputs for the models. While the features chosen gave the best performance, it is not conclusive that they are the optimal set of features, since many other features are still uninvestigated.

For energy prediction of 5G base stations, this thesis finds that using a more balanced dataset, in terms of the number of samples for each product, has a positive impact for the ANN and the Gradient Boosted Trees model while the linear regression performs worse. Having more balanced data over more features may have similar impact on model performance. Utilization features such as PRB DL utilization and RB symbol utilization are obvious features that can be improved. As seen in Figure 4.22 there are few samples for higher utilization compared to samples for lower utilization. It is reasonable to assume that, by sampling more data for higher utilization, the model performances will improve. Although the results look promising we believe that more work will prove to be more beneficial in the field of energy consumption modelling and in the future open up

more opportunities for energy consumption optimization.

References

[1] 5G vs 4G. https://www.ericsson.com/en/5g/5g-vs-4g. Accessed: 2023-04-30.

[2] IoT - så funkar det. https://iotsverige.se/om-oss/iot-sa-funkar-det. Accessed: 2023-05-01.

[3] L. Bottou et al. Stochastic gradient learning in neural networks. *Proceedings of Neuro-Nımes*, 91(8):12, 1991.

[4] E. Dahlman, S. Parkvall, and J. Sköld. *5G NR - The Next Generation Wireless Access Technology*. Academic Press, 1st edition, 2018.

[5] Energimyndigheten. Nuläget på elmarknaden. Technical report, January 2023.

[6] P. Frenger. OFDM MIMO: 4G Long Term Evolution, 5G New Radio, and 6G. [Internal PowerPoint presentation], 2021.

[7] P. Frenger. RDI-RAFT. [Internal PowerPoint presentation], 2021.

[8] Y. Freund and R. Schapire. Large margin classification using the perceptron algorithm. *Machine Learning*, 37, 02 1999.

[9] J. B. G. Wright. Definition: Orthogonal frequency-division multiplexing (ofdm). https://www.techtarget.com/searchnetworking/definition/orthogonal-frequency-division-multiplexing. Accessed: 2023-04-27.

[10] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2017.

[11] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6(6):861–867, 1993.

[12] H. Li, B. Chen, J. Liang, B. Shen, and C. Shen. Application of energy consumption model and energy conservation technology in new infrastructure. In *2022 IEEE 5th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, volume 5, pages 1638–1644, 2022.

[13] A. Mughees, M. Tahir, M. A. Sheikh, and A. Ahad. Towards energy efficient 5g networks using machine learning: Taxonomy, research challenges, and future research directions. *IEEE Access*, 8:187498–187522, 2020.

[14] M. A. Nielsen. *Neural Networks and Deep Learning*. Determination Press, 2015.

[15] N. Piovesan, D. Lopez-Perez, A. D. Domenico, X. Geng, H. Bao, and M. Debbah. Machine learning and analytical power consumption models for 5G base stations. *IEEE Communications Magazine*, 60(10):56–62, oct 2022.

[16] C. B. Pop, V. R. Chifu, C. Cordea, E. S. Chifu, and O. Barsan. Forecasting the short-term energy consumption using random forests and gradient boosting. In *2021 20th RoEduNet Conference: Networking in Education and Research (RoEduNet)*, pages 1–6, 2021.

[17] A. L. Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3):210–229, 1959.

[18] J. Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

[19] O. Shurdi, L. Ruci, A. Biberaj, and G. Mesi. 5G energy efficiency overview. *European Scientific Journal*, 17(3):315–27, 2021.

[20] E. K. T. Hatt, P. Jarich. Radar: The sustainable telco. Technical report, GSMA Intelligence, 12 2021.

[21] R. D. A. A. J. F. A. R. Thomas Berglund, Pål Frenger. Energy Performance Observability. [Internal document], 2023.

[22] Z. Wu and W. Chu. Sampling strategy analysis of machine learning models for energy consumption prediction. In *2021 IEEE 9th International Conference on Smart Energy Grid Engineering (SEGE)*, pages 77–81, 2021.

[23] Z. Y. Z.R. Yang. Comprehensive biomedical physics. 2014.

Appendix A

Table 6.1: MSE and MAE for all product evaluations with the ANN model on Dataset 2.

Product	MSE $[(\text{EU})^2 \cdot 10^{-4}]$	MAE $[\text{EU} \cdot 10^{-3}]$
1	**0.6072**	5.3540
2	2.0468	9.3104
3	1.7079	8.9811
4	1.9709	10.8233
5	5.0943	17.6481
6	4.4835	16.3088
7	1.2296	6.6507
8	0.7253	6.9830
9	1.5384	9.3178
10	2.2055	10.0334
11	0.8451	5.8851
12	0.6895	**5.0241**

Table 6.2: MSE and MAE for all product evaluations with the gradient boosted trees model on Dataset 2.

Product	MSE $[(\text{EU})^2 \cdot 10^{-4}]$	MAE $[\text{EU} \cdot 10^{-3}]$
1	0.6858	5.731
2	2.2310	8.967
3	1.7936	8.866
4	2.6337	12.156
5	6.8356	21.266
6	4.9051	17.222
7	1.4337	8.248
8	**0.5076**	**5.700**
9	1.9075	10.112
10	2.8645	12.777
11	0.6147	5.789
12	0.9428	6.372

$$y =$$

Table 6.3: Data

Device	**Value 1**	**Value 2**
AIR 3239 B78C	4.9677970790548046e-05	0.004939646534424238
AIR 6488 B43	0.00015105145296995598	0.008099466483947148
Radio 2217 B1	0.00011551563903547389	0.0071623534678615245
Radio 2238 B8 B20 B28B	9.861535423137797e-05	0.007303589611979615
Radio 2242 B1 B3	6.982869189497843e-05	0.004594543495550992
Radio 2460 24B8 24B20 24B28B M01	0.0002396098884717634	0.011308742841456347
Radio 4415 B1	4.965182231566099e-05	0.003881556003421639
Radio 4422 B78C	1.8882490802764854e-05	0.002889416381459159
Radio 4442 B1 B3	7.929654095630804e-05	0.006250891646642039
Radio 4480 44B1 44B3 C	9.734201230296906e-05	0.006333066797995157
Radio 8823 B43	3.731275873348221e-05	0.00460499034848658
Radio 8863 B78K	2.4903416978329708e-05	0.003564498767982859

Table 6.4: MSE and MAE for all product evaluations with the linear regression model on Dataset 2.

Product	MSE $[(\text{EU})^2 \cdot 10^{-4}]$	MAE $[\text{EU} \cdot 10^{-3}]$
1	1.6380	8.6964
2	**0.6186**	**4.8085**
3	4.3846	14.2534
4	1.2857	8.6512
5	9.2965	24.2374
6	2.6483	12.2871
7	6.2563	15.1661
8	8.6411	23.5234
9	3.3609	14.0638
10	6.2444	19.7632
11	3.2273	11.3263
12	0.6954	5.1934

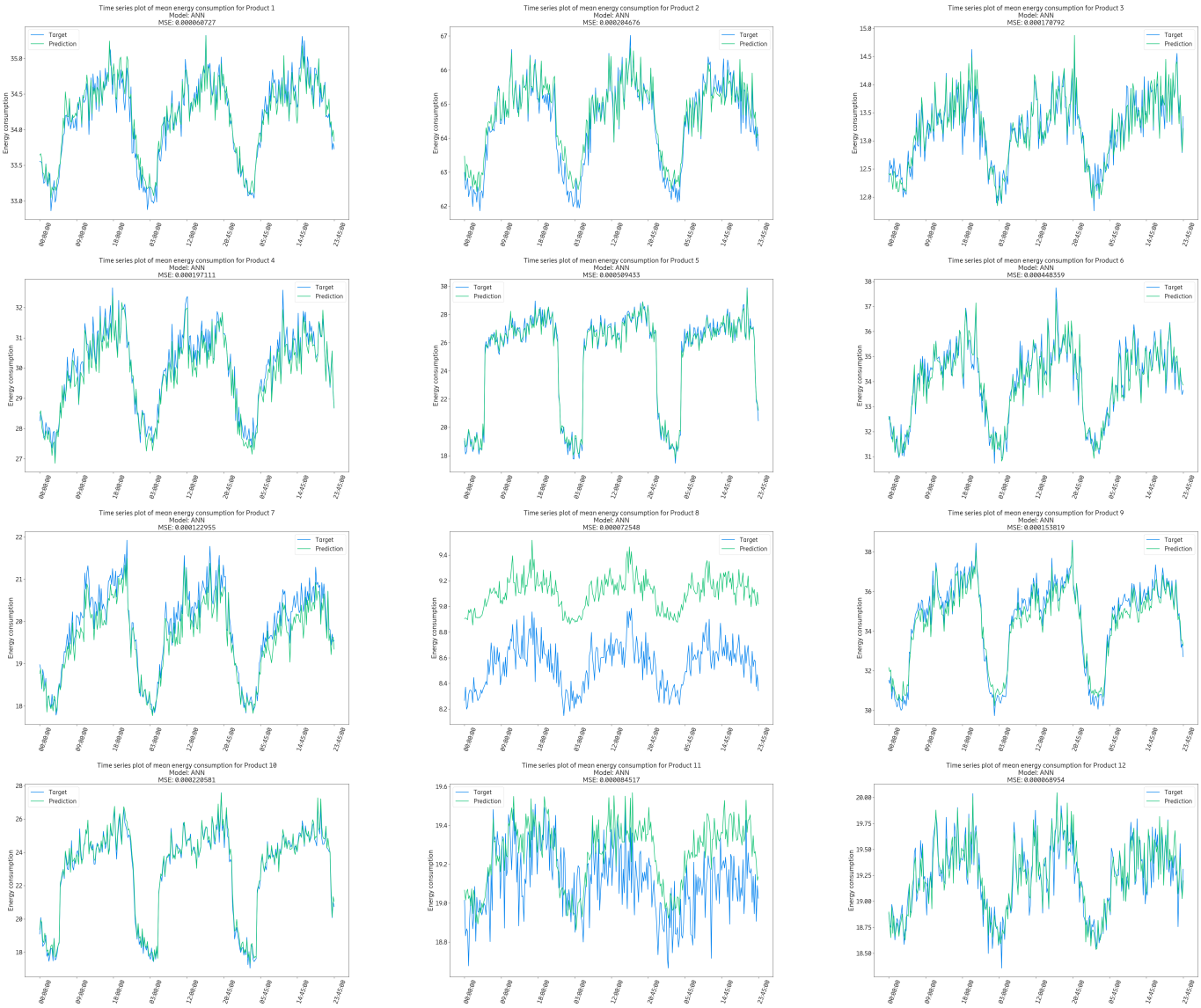


Figure 6.1: Time series plot for all products in Dataset 2 using the ANN model.

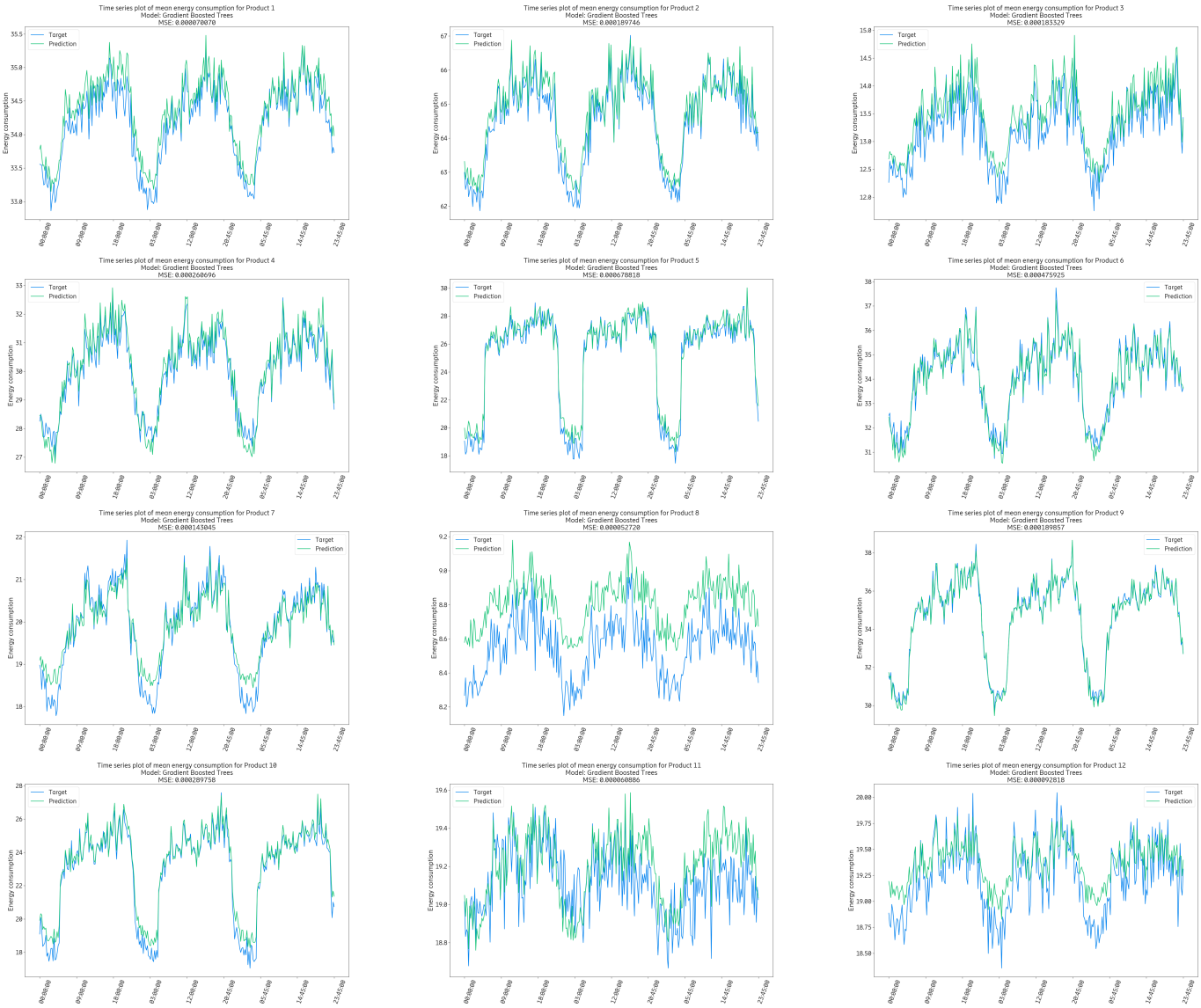


Figure 6.2: Time series plot for all products in Dataset 2 using the gradient boosted trees model.

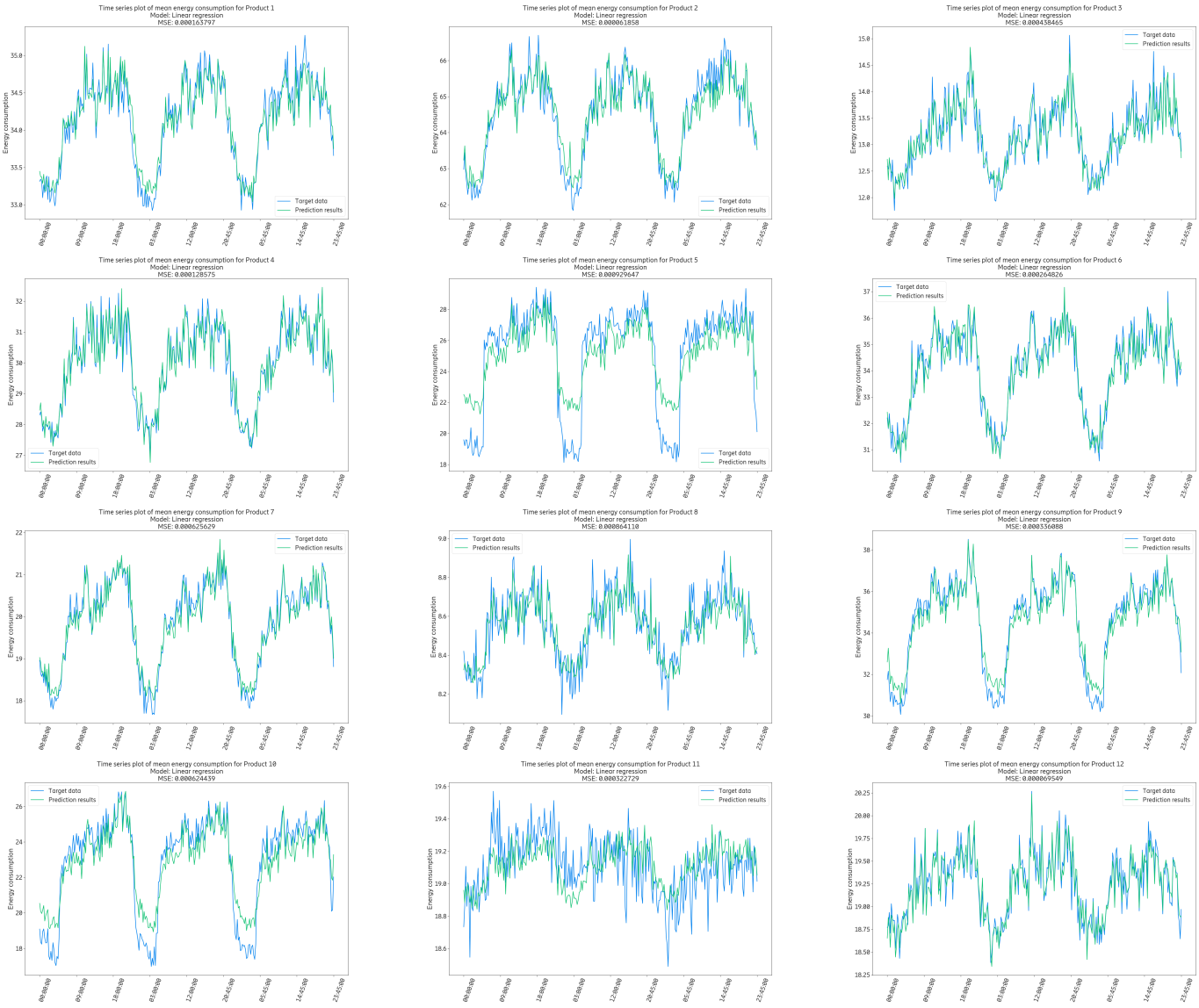


Figure 6.3: Time series plot for all products in Dataset 2 using the linear regression model.