Machine Learning Based Threshold Tuning for Optimal Massive MIMO Performance

Ebba Lundgren and Nora Olofsson

Department of Electrical and Information Technology
Lund University

Supervisor: Fredrik Tufvesson, Harish Venkatraman Bhat

Examiner: Michael Lentmaier

June 1, 2026

Printed in Sweden
Tryckeriet i E-huset, Lund

Abstract

This thesis explores how machine learning can be applied to dynamically set the optimal threshold that determines if a user should be granted Sounding Reference Signals (SRS) resources at cell level. The goal of the thesis is to investigate whether this approach can improve the downlink performance on cell level compared to a static set threshold. Collected network measurements were used to train and evaluate different machine learning models for estimating spectral efficiency across different threshold configurations. Among the two evaluated approaches, XGBoost demonstrated the highest performance. The analysis revealed that features related to long-term network behavior, including the mean User Equipment (UE) SRS Signal-to-Noise Ratio (SINR), the mean path loss of the uplink and the aggregated throughput, contributed the most to the predictions, while short-term variability, the variance of the same parameters, showed less influence. The proposed adaptive threshold selection approach performed better than fixed threshold configurations in 48.3% of the evaluated cases, with a net value of around 0.15 bps/Hz, suggesting that dynamic optimization can improve network performance. At the same time, the thesis identified challenges associated with data quality, overlap between threshold configurations, and disproportion of the data set, which limited the stability and generalization of the model. The findings indicate that machine learning based dynamic threshold optimization is a promising direction for wireless network management and motivates further research on improved data collection and model development.

Keywords: Massive MIMO, Machine Learning, Supervised Learning, Random Forest, XGBoost, SRS, Dynamic Threshold, Cell Level, Spectral Efficiency.

Acknowledgements

We would like to express our sincere gratitude to the team at Ericsson for making this thesis possible. We are especially thankful to Harish Venkatraman Bhat, Arlind Iseni, José Maria Garcia Perera, Juan Guirado Lopez-puigcerver, David Lundberg and Magnus Jönsson for their kindness, support and knowledge throughout the process. Your willingness to take the time to guide us, answer our questions, and help us along the way has meant a great deal to us.

We would also like to extend our appreciation and gratitude to Fredrik Tufvesson for the professional guidance and support provided during this thesis work. We would also like to express our gratitude to Michael Lentmaier.

Lastly, we would like to thank our families and loved ones for their endless support and encouragement throughout these five years at LTH. We are deeply grateful for everything you have done for us.

Popular Science Summary

Wireless communication has become an essential part of everyday life. This thesis investigates whether machine learning can be used to improve downlink performance in Massive MIMO systems.

What exactly does Massive MIMO mean? In Massive MIMO, which stands for Multiple-Input Multiple-Output, the basestation is equipped with multiple antennas. They can be used to serve both single users, called SU-MIMO, and multiple users simultaneously, called MU-MIMO. Massive MIMO enables higher speed and throughput for User Equipments (also called UEs). In Massive MIMO, the base station uses either reciprocity-based beamforming or codebook-based beamforming to direct signals toward specific UEs. Reciprocity is often more desirable than codebook, since UEs can achieve higher speeds due to the extended channel information. Reciprocity-based beamforming relies on Sounding Reference Signals (SRS), which allow the base station to estimate the uplink channel. So, why not use reciprocity at all times? SRS resources are limited and cannot be assigned to every UE at the same time, since this would introduce interference and destroy the good connection. To decide which users should receive SRS resources and therefore also reciprocity, networks today use thresholds of the UEs SRS SINR. SINR measures how strong a wireless signal is compared to interference and background noise.

The baseline for this thesis, is that the threshold is fixed on node level and does not adapt to changing network conditions. As a result, some users who could benefit from SRS may not receive access to it. This thesis explores whether a machine learning model can dynamically adjust this threshold based on predicted spectral efficiency and, if so, whether such an approach can improve overall performance. By collecting, cleaning, and analyzing network data, a machine learning model was developed and evaluated. The results showed improved downlink spectral efficiency in 48.3% of the evaluated cases compared to using fixed thresholds. These findings indicate that adaptive thresholding using machine learning has the potential to improve wireless network performance and enable more efficient resource allocation. The positive net value of around 0.15 bps/Hz also indicates that the dynamic model successfully outperforms the static threshold. In practice, this has the potential to achieve better network performance for users.

Table of Contents

Chapter 1
Introduction

This chapter describes the fundamentals of massive MIMO, reviews relevant previous work, and defines the research questions. This chapter also includes limitations and contribution of the thesis.

1.1 Motivation

In the modern world, reliable internet connectivity is fundamental. As global data consumption continues to grow, the transition from 5G New Radio (NR) to future 6G technologies is driven by the increasing demand for connectivity. This rapid growth places strict requirements on wireless networks in terms of capacity, efficiency and user density. One of the main challenges is the ability to serve multiple users simultaneously within the same frequency resources.

To address this challenge, Massive MIMO (Multiple-Input Multiple-Output) serves as a key enabling technology. Its main purpose is to improve the efficiency of wireless networks by utilizing a large number of antennas at the base station, enabling both Single-User MIMO (SU-MIMO), where one user benefits from multiple streams, and Multi-User MIMO (MU-MIMO), where multiple users can be served concurrently on the same frequency.

Unlike traditional systems, that broadcast signals in all directions, Massive MIMO uses beamforming to focus transmissions into narrow, directed beams aimed at specific user equipments (UEs). This targeted transmission enables the benefits of Massive MIMO, improves signal strength and reduces interference [1].

1.1.1 Downlink Performance Measurements

The downlink performance, i.e. the quality at which data is transferred from the base station to the UE, can be measured in several ways. In this thesis, Block Error Rate (BLER), Throughput and Spectral Efficiency (SE) will be considered when measuring this. BLER measures the ratio of incorrectly received data to the total transmitted data, usually expressed as a percentage. Throughput is the rate at which data is successfully delivered from the base station to the UE per unit of time. A high throughput indicates good conditions for the downlink. Lastly, SE measures how efficiently a given bandwidth is used to transmit data. A high SE

indicates that the downlink performance is good. It can be increased in multiple ways, for example through beamforming, nullforming and MIMO [2].

1.1.2 Dynamically Setting the Threshold

To improve the downlink performance further, the network uses Sounding Reference Signals (SRS) to estimate the channel condition. This allows for reciprocity-based beamforming, which can provide better channel information and therefore better performance than codebook-based beamforming for Massive MIMO [2]. More about these two different methods can be read about in Chapter 2.

SRS resources are limited and cannot be allocated to all UEs simultaneously. Therefore, a threshold is introduced to select which UEs are granted SRS resources. If the threshold is set too low, many UEs will be assigned SRS resources. This leads to increased interference to neighboring cells and reduced performance. At the same time, if the threshold is too high, there will be missed opportunities to use SRS for the UEs. It is also important to take back SRS resources from UEs that no longer need them. This is done using an additional threshold that decides when allocated SRS resources should be removed. In this thesis, the first threshold is the one that will be looked into, i.e., SRS allocation.

This threshold is based on SRS SINR (Signal-to-Interference-plus-Noise Ratio based on SRS). This is a measurement of the uplink channel quality estimated by the base station using the received SRS signals. For this thesis, the baseline is a static threshold, meaning that it remains constant within a given network and does not adapt to variations in network conditions or traffic dynamics. This static configuration may limit the system's ability to respond efficiently to changing network environments. Therefore, it is reasonable to hypothesize that downlink performance could be improved by introducing a dynamically adjusted threshold that adapts to these changing network parameters. Specifically, this thesis investigates whether a dynamically adjusted threshold can improve downlink performance in terms of cell level SE. This forms the primary motivation for this study.

1.2 Previous Work

Previous research has been conducted in this area, and this thesis mainly builds upon two earlier master theses.

Firstly, a thesis by Johan Skäremo at Malmö Universitet [3]. This thesis approached SRS allocation from a UE-level perspective, using a machine learning based method to decide whether an individual UE should be assigned SRS or not. The goal of the research was to estimate the Spectral Efficiency for both precoding and reciprocity-based precoding. This helps determine if a user would benefit from being assigned SRS resources when using reciprocity-based precoding. The decision was primarily evaluated from a performance perspective, where throughput was used. However, the threshold itself was not explicitly analyzed or tuned. Secondly, a master thesis by Arlind Iseni at Blekinge Tekniska Högskola [4], which investigates using Machine Learning (ML) to automate the selection between codebook-based and reciprocity-based precoding in 5G massive MIMO

networks. The way these two differ is that the first thesis by Skäremo [3], estimates SE under different precoding types to determine whether a user would benefit from SRS resources, while the second thesis by Iseni [4], estimates the SE to choose between the two precoding techniques.

Furthermore, Chaki et al. [5] investigated how different SRS configurations impact 5G performance, particularly when the channel changes over time (channel aging). They demonstrated a clear trade-off: measuring the entire frequency band at once provides the most accurate data but requires significant power from the user device. Conversely, measuring the band in smaller pieces saves power but introduces errors since the channel information becomes outdated between measurements. This work confirms that static allocation strategies are inefficient and that the network must balance accuracy against aging effects to maintain optimal performance. Since Chaki et al. demonstrated that no single static configuration is optimal for all scenarios, there is a clear need for a more adaptive solution. This limitation directly motivates this thesis, which aims to develop an ML model to dynamically estimate the optimal SRS SINR threshold.

Lastly, addressing the limitations of periodic SRS allocation, Fiandrino et al. [6] argued that fixed schedules are inefficient because they can not adapt to irregular user traffic. To solve this, they proposed a machine learning framework called TRADER, which uses deep learning to predict exactly when a user will transmit data. By forecasting these traffic bursts, the system can dynamically trigger an SRS transmission just in time, ensuring the channel information is up to date without wasting resources. Their experiments confirmed that this data-driven approach significantly outperforms standard static allocation schemes, resulting in better channel quality and higher throughput.

Finally, while Fiandrino et al. [2] successfully demonstrated the potential of machine learning for optimizing SRS scheduling based on traffic patterns, their work primarily focused on the timing of transmissions.

1.3 Purpose and Aims

The main purpose of this thesis is to investigate whether the allocation of SRS resources on cell level can be improved by utilizing an ML model to propose an adaptive threshold instead of using a static one. The process involves identifying parameters that are strongly associated with SE, considering both positive and negative correlations. These parameters are then used as input features for training the ML model. The model is evaluated on the basis of its ability to effectively predict SE for different network contexts. The final solution includes both the ML model and the threshold algorithm, and is evaluated based on its overall performance as a function for selecting SRS thresholds that result in higher SE.

1.3.1 Research Questions

RQ1: How can a machine learning model be designed to dynamically determine the optimal SRS SINR threshold at a cell level?

RQ2: Which input parameters have the highest impact when predicting the optimal SRS threshold?

RQ3: How does a dynamic threshold affect downlink performance compared to fixed thresholds?

1.3.2 Limitations

This thesis is limited to the investigation of SU-MIMO systems. MU-MIMO are outside the scope of this study. Another limitation is the time frame of the thesis. The thesis is conducted over a period of 20 weeks, which restricts the depth of the investigation. Data availability also sets a limitation. Due to time and resource constraints, it is not possible to access or analyze data representing all possible real world scenarios. However, efforts have been made to use relevant and representative datasets to ensure meaningful and reliable results.

1.3.3 Contribution

This thesis contributes to the field by analyzing and preprocessing data obtained from real world scenarios. The primary objective is to identify the most important parameters related to SE on cell level. This is done by evaluating if a dynamically set threshold can improve this SE value. The results aim to contribute to a further understanding of the allocation mechanisms and providing new insights that will support future research and practical implementations.

Chapter 2
Background

This chapter provides the necessary background to understand cellular networks, Massive MIMO and different precoding techniques. It provides the information needed to understand the scope of the research. It also includes how the literature research was conducted.

2.1 Cellular Networks Fundamentals

A mobile network consists of several base stations that send and receive signals [7]. The served area is divided into cells, which each are served by a base station. The base stations handle both uplink and downlink data for UEs within their own cell. The coverage area of each cell varies depending on different factors, such as the used frequency band, output power, among others [8]. A base station has multiple antennas that cover different directions. Each antenna covers their own local section, which is one part of the area around the base station. For the UE to be able to connect to the base station, it needs to be within the range of one of these sections [7].

Different mobile systems operate on different frequency bands in the radio spectrum. Each network operator is assigned a specific part of the spectrum, which is then divided into smaller channels. This enables multiple UEs to access the same spectrum simultaneously, as they use separate frequency bands or time periods to avoid interference [8]. Spectrum is a scarce and expensive resource for operators, which is why spectrum efficiency is of great importance and why this thesis aims to maximize it. From these base stations, logs of data can be retrieved, containing information of the UEs that has connected to them. These logs contain information such as the transmission time, the user involved, the precoder that was used and other related parameters. These are the logs used in this master's thesis. These will be further described in Chapter 4.

2.1.1 Introduction to 5G

5G represents the fifth generation of cellular networks and is a further development of the previous systems, such as 4G and 3G. 5G enables considerably higher data rates, reduced latency, improved positioning accuracy, and the ability to support a significantly larger number of connected devices simultaneously.

These improvements are primarily achieved through multiple antennas, new access methods, higher bandwidths and the use of new frequency bands (from around 3.5 GHz to 28-39 GHz) in additional parts of the radio spectrum, while still utilizing established frequencies from earlier generations. By integrating low, mid, and high-band spectrum, the network can balance coverage, capacity, and performance more efficiently [9].

2.1.2 Massive MIMO

Massive MIMO is one of the fundamental technologies in modern 5G networks. By equipping base stations with a large number of antennas, the network can serve more users simultaneously, increase individual user throughput, and improve overall coverage. These performance gains are achieved through techniques within massive MIMO called beamforming, nullforming, and spatial multiplexing (commonly called MIMO) [2].

Beamforming

Beamforming is a technique in MIMO that amplifies signals in specific directions instead of broadcasting it equally in all directions. By directing the signal toward the target UE, the received signal power is increased, which improves the Signal-to-Interference-plus-Noise Ratio (SINR). Improving the SINR leads to better conditions for this UE, in terms of coverage, throughput and capacity [2].

Nullforming

Nullforming is closely related to beamforming but serves the opposite purpose. Instead of strengthening the signal in a desired direction, nullforming reduces or suppresses signal power toward specific directions. This is typically done to minimize interference toward certain UEs while serving another UE. By placing "nulls" in the radiation pattern toward interfering users, the system can improve overall SINR and enhance system performance, especially in dense networks [2].

Spatial Multiplexing

Spatial multiplexing allows the base station and the UE to send multiple data streams at the same time using the same frequency resources. This is possible because both sides use multiple antennas, which makes it possible to separate the signals in space. In other words, the antennas create different paths for the data streams so they do not interfere with each other. By transmitting several streams simultaneously, the overall data rate can be increased without using more bandwidth [2].

SU-MUMO vs MU-MIMO

Massive MIMO systems can generally be categorized into Single-User MIMO (SU-MIMO) and Multi-User MIMO (MU-MIMO). SU-MIMO and MU-MIMO are two forms of spatial multiplexing. In SU-MIMO, the base station sends multiple data

streams to one single UE at the same time. The goal is to increase the data rate for that specific, single user by using the multiple antennas to create separate signal paths. In MU-MIMO, the base station serves several UEs at the same time using the same time and frequency resources. Apart from separating streams for one user, the system separates different users in space. This allows the network to support more users at once and improves the overall capacity and efficiency of the cell [2].

2.1.3 Channel State Information (CSI)

For Massive MIMO to function properly, the base station must have accurate knowledge of the radio channel between itself and the user. This channel information is often referred to as Channel State Information (CSI), and it is essential for creating directed beams and separating users in the spatial domain. Without reliable CSI, the base station can not efficiently focus the transmitted signals or control interference [10].

2.2 Precoding in Massive MIMO

Based on channel information, the base station applies precoding. Precoding is a technique used at the transmitter to control how the signal is sent out. Instead of transmitting the signal equally in all directions, the system adjusts it so that more power is sent toward a specific user. In wireless networks, precoders are used to focus strong signal beams on the intended user while at the same time reducing interference to other users [11]. The performance of Massive MIMO therefore depends on how accurately the precoding reflects the actual channel conditions. In this master thesis, two precoding approaches will be discussed: codebook-based precoding and reciprocity-based precoding.

2.2.1 Codebook-Based Precoding

In the codebook-based approach, the base station transmits a Reference Signal (RS) over the downlink (DL) channel. The UE measures this signal in order to evaluate the channel conditions. The UE is provided with a predefined precoder codebook, specified by the 3GPP standard, which contains a set of possible beamforming configurations. Based on its measurements of the downlink channel, the UE selects the beam from the codebook that best matches the current channel conditions. The UE then feeds this information back to the base station, typically in the form of a precoding matrix indicator (PMI). The base station uses this recommended beamformer when transmitting data to the UE [2]. One advantage of this method is that it works well in Frequency Division Duplexing (FDD) systems, where uplink and downlink operate on different frequency bands. However, since the base station only selects from a limited set of predefined beamforming options, the precision is restricted by the size and design of the codebook.

2.2.2 Reciprocity-Based Precoding

In the reciprocity-based approach, the UE transmits an uplink (UL) SRS to the base station. In Time Division Duplexing (TDD) systems, the same frequency band is used for both uplink and downlink, but at different times. Because of this, the radio channel can be seen as reciprocal, meaning that the channel characteristics are essentially the same in both directions within a short time interval. By receiving the SRS, the base station can directly estimate the uplink channel and use this estimate to determine the downlink channel state. This provides the base station with the channel information without requiring explicit feedback from the UE. The main advantage of reciprocity-based precoding is that the transmitter obtains more complete and accurate channel information compared to the codebook-based approach. This allows for more precise beamforming, where the beam can be highly concentrated toward the intended UE. In addition, the detailed channel knowledge enables effective nullforming, meaning that interference toward other UEs can be significantly reduced. However, this method has certain limitations. Since the channel estimation relies on the received SRS, the uplink signal must be of sufficient quality. At longer distances or in poor radio conditions, the accuracy of the channel estimate decreases, which directly impacts the effectiveness of beamforming and nullforming. In these situations, codebook-based precoding is used as a fallback. [2].

2.3 Literature Research

To identify relevant literature for this thesis, a literature review was conducted. Scientific articles, books, and conference papers related to the research topic were collected through academic databases and search engines, for example IEEE and Lunds University Libraries Search Tool, called Finn. Keywords, such as "Cellular networks", "Massive MIMO", "SRS SINR", and other relevant words to the study were used to locate sources addressing similar problems, methodologies, and findings. The selected literature was evaluated based on its relevance, credibility, and publication date to ensure that the sources were reliable and mostly recent information was used, considering this is a field that is constantly developing. Snowballing was also used as a literature search strategy. Specifically, backward snowballing was applied by reviewing the reference lists of selected articles to identify additional relevant studies [12].

Chapter 3
Methodology

For this part, the methodology used for this master thesis will be introduced and discussed. This includes the data collection, challenges, preprocessing, the different approaches for the solution and an overview of the final dataframe and network parameters. The data were obtained from a real live network and consists of network traces using different SRS SINR thresholds. In total, five different thresholds were used. In this thesis, they are referred to as Threshold A, B, C, D, and E, where Threshold A represents the lowest threshold and Threshold E the highest.

3.1 Method Selection and Approaches

During the course of the thesis, the approach evolved.

3.1.1 Initial Approach

Initially, the idea was to analyze data using a single threshold, specifically the lowest one, Threshold A. The dataset was also toggled between codebook and reciprocity modes to distinguish between cases where codebook was active and where reciprocity was active. In times were a UE is granted SRS resources, reciprocity is enabled. However, a UE does not always use reciprocity in these cases. If necessary and more useful, the codebook is used as a fallback. This results in codebook "being available" at all times, whereas reciprocity was only enabled during specific time intervals.

By setting the threshold to a very low value, the intention was to identify a crossing point between codebook and reciprocity in terms of Spectral Efficiency. The hypothesis was that codebook would perform better at lower SRS SINR values, and that beyond a certain SRS SINR level, reciprocity would outperform codebook. This transition point was expected to represent the optimal threshold. The idea behind this analysis was to observe for different cells and times of the day, if there was a possibility to predict an optimal threshold dynamically. A benefit to this approach is that it used a continuous value space, which means the optimal crossing point could be found anywhere in the range. This makes the approach flexible as it allows for a more precise estimate of the true crossing point, instead of being limited to fixed thresholds. However, when analyzing the resulting plots, see Figure

4.4 in Chapter 4, no clear optimal threshold could be identified. Instead, multiple crossing points were observed at different SRS SINR values. Additionally, the performance distributions of codebook and reciprocity differed more than initially expected. This inconsistency may be because of factors such as varying network conditions. The presence of multiple crossing points indicated that the hypothesis of just one crossing point was not valid, so it could not be further explored. Based on this insight, the work proceeded to the second approach.

3.1.2 Evolved Approach

For the second approach, which focuses on tuning the threshold using ML, a new type of dataset was required. This dataset included multiple threshold levels, ranging from Threshold A to Threshold E, to provide sufficient variability for training the model. The objective was to train the model on a diverse set of thresholds while also including the currently deployed static threshold. This enabled a comparison between the existing solution and the proposed ML based approach. Practical limitations related to company resources and project time constraints restricted the number of thresholds and, as a result, only five distinct thresholds were used for training and evaluation. Even though this covers a good spread of thresholds, it limits the optimal solution since values between the predefined thresholds cannot be directly represented or selected. A challenge with the network data was the lack of "ground truth" labels, as it is impossible to know which threshold would have been optimal for a specific situation beforehand. To navigate these missing labels, the approach focuses on predicting the spectral efficiency for each threshold instead of directly estimating the best threshold. This became the final approach used in the thesis.

3.2 Data Collection

For this thesis to be feasible, a substantial amount of data was needed. The data used in this thesis was collected at Ericsson from a real live network. It was collected with the support of the team and is based on observations from a real live network. Data collection was conducted multiple times during the entire scope of the thesis under operational conditions to ensure reliability and correctness of the data. Established procedures were followed to maintain consistency and data quality throughout the process.

3.2.1 Challenges with the Data Collection

The primary objective was to retrieve data across multiple time points in order to develop a more nuanced and comprehensive representation. However, several issues emerged during the data collection process. In some cases, unintended settings when the data was ordered were introduced, which compromised the quality and reliability of the data. Another recurring issue was that in the datasets collected, not all thresholds were present. One approach to navigate this, was to concatenate data with different thresholds from multiple days. However, this is not ideal, since the data comes from different time points and contexts. This means that each

threshold may reflect different conditions, which can introduce bias when training the machine learning model.

Other issues were caused by changes affecting the data collection process. Initially, threshold data were separated by log, making each threshold distinct from the others. However, at one point during the process of data collection, this approach was no longer feasible due to a software upgrade in the real live network. Instead, all thresholds appeared in every collected log. This occurred because Threshold A represents the lowest threshold value, meaning that when data was collected for Threshold A, all other threshold values above it were also within the allowed range and therefore implicitly included as well. As a result, the first log contained all thresholds, the second contained four, and so on. This led to a very limited number of data points for the intended threshold in each case.

To address this issue, a revised approach was implemented in which data collection was conducted in reverse order. Specifically, collection began with Threshold E (the highest) and proceeded down to Threshold A (the lowest). The idea was that when collecting data for Threshold E, since it is the highest threshold value, no unintended UEs with lower SRS SINR values would be included in this set. This is because the threshold is set to a higher value, meaning that only UEs within this higher range are allowed to receive SRS. Ideally, Log 1 would contain only Threshold E, Log 2 would contain Thresholds E and D, and so forth. Only Thresholds A, B, D and E was collected due to real live network constraints. This resulted in the final dataset containing four logs, corresponding to Thresholds A, B, D, and E, while no dedicated log was collected specifically for Threshold C. However, in practice, Threshold C appeared in all logs, even in cases where it was outside the intended range for higher thresholds such as Threshold D and Threshold E. For example, in Log 1, both Threshold E and Threshold C were observed. This occurred because some UEs had already established a connection using Threshold C before the data collection started and continued maintaining that connection during the capture, causing Threshold C to remain present in the logs.

The reason for this behavior is that the threshold configuration is applied when the UE connection is established and can not be changed at runtime for already connected users. As a result, UEs connected with Threshold C before the start of the data collection kept their existing threshold configuration until the SRS resource was released. Since no dedicated log was collected specifically for Threshold C, samples corresponding to Threshold C were instead extracted from Log 1, which was originally collected for Threshold E, to ensure that Threshold C was included during model training. Despite these complications, the final dataset still allowed for effective filtering of the intended thresholds in each log, as the target threshold remained dominant. More about these challenges are discussed in Chapter 6.

3.3 Dataset Overview

The data was analyzed using Jupyter notebooks together with Python libraries such as Pandas and Polars. The data logs consist of multiple traces, where each

trace represents a continuous recording of network activity over time. These traces capture different types of transmission slots such as downlink, uplink, and special slots, allowing the dataset to reflect a wide range of network conditions. These traces are loaded into dataframes, which are data structures with rows and columns. Each row represents one observation and each column represents a feature or measurement.

3.4 Preprocessing the Data

Data preprocessing involves cleaning the dataset to ensure it is accurate and reliable. This means identifying and fixing errors, for example missing values, outliers, and incomplete information. Since an ML model's performance depends on the quality of the input, this step is important. The better and more accurate the data is, the more accurate the results will be [13]. Therefore, preprocessing was a central part of this thesis which was always done when retrieving new data. During data preprocessing, null values were identified and handled to ensure that the functions used worked correctly. Rows containing null or Not a Number (NaN) values were removed to ensure that the final dataset consisted only of complete and usable observations. Null values do not necessarily indicate incorrect data but it means that data was missing for a specific time.

3.4.1 Filtering

After cleaning the dataset by removing invalid values, it was further filtered to retain only the relevant data. The idea was to filter the data to keep only the data that used reciprocity as much as possible and ignore transmissions that only act as noise in this scenario. These were some of the filtering steps that were applied:

- **Precoder and MIMO Type:** This filtering was done in order to make sure only SU Codebook and SU Reciprocity was used.
- **Carrier Aggregation:** This filtering was applied to exclude carrier aggregation (CA), where a UE can utilize multiple cells simultaneously to increase data rates [2]. By applying this filter, only data from the Primary Cell is retained, while contributions from other cells are removed.
- **Reciprocity On:** This is an approximated parameter for when reciprocity was on. The inclusion of this parameter in the filtering process was necessary due to the toggling nature of the dataset. Periods where reciprocity was not allowed (i.e., when the UE did not have SRS configured) were excluded, since codebook was then always used. These situations only introduced noise into the analysis, as the focus was on reciprocity UEs.

  It should be noted that the dataset includes both SRS and non-SRS users. However, to avoid bias towards threshold E (the highest threshold), which corresponds to the most favorable Radio Frequency (RF), i.e. the condition between the UE and basestation, both types of users were retained while ensuring consistent filtering across conditions.

Although reciprocity is enabled in the remaining dataset, fallback to codebook may still occur, and some UEs may operate without SRS. The dataset therefore contains a mixture of these scenarios. Nevertheless, time periods where only codebook was permitted were removed, as they contribute to noise in the analysis.

- **The given Threshold:** The intended threshold, selected from the discrete set of predefined options (A–E). This was because other unintended thresholds were present due to long UE connections.

These filtering criteria was developed together with the team at Ericsson.

3.4.2 Network Parameters

When combined, the traces form a comprehensive dataset containing a variety of network parameters. Together, they provide a diverse set of radio condition parameters. Given the large number of available parameters in the original dataset, a selection step was necessary to determine which features should be used as input to the ML model. The selection was based on a previously established subset of features identified in earlier work [3]. This subset, consisting of 26 parameters, served as a starting point for the analysis. In addition, the selection process was guided by domain knowledge. Input from experts at Ericsson helped ensure that the chosen features are relevant from a practical network perspective and are known to influence SE, as well as aligning with the objectives of this thesis.

Since this thesis investigates whether the SRS SINR threshold can be defined at the *cell level*, the selected parameters needed to be represented accordingly. However, the collected data are originally measured at the UE level rather than the cell level. Therefore, the parameters were aggregated and dynamically grouped to reflect cell level characteristics. This allowed the data to be partitioned into 500 ms buckets based on the "time" index, while simultaneously grouping by categories. These categories were: cell id, name of the logs and the given threshold. An offset of -500 ms was also included to prevent data leakage where the model could potentially access parameter values before they occurred in real time.

Aggregating the parameters required defining appropriate aggregation methods for each metric. For example, SE was computed across all UEs within a cell, rather than considering individual UE values. Similarly, for path loss, both the mean and the variance were calculated. This allows for analyzing not only the average radio conditions but also the variability within the cell, and how these factors jointly influence SE. The selected parameters, along with their corresponding cell level aggregations, are listed below:

- **Spectral Efficiency:** Computed as the SE across all UEs within the cell, representing the overall cell performance. SE on cell level is approximated as follows:

$$\eta = \left(\frac{\sum_{i=0}^{n} (f_i \cdot S_i)}{\sum_{i=0}^{n} R_i} \right) \cdot 8, \text{bits/resource element}$$

  - η : Spectral Efficiency (bits per resource element)

- f_i : Indicator variable (new data flag), equals 1 if new data is transmitted, otherwise 0.
- S_i : Size of transmitted data in bytes.
- R_i : Number of allocated resources.
- $\sum$: Summation over all observations i
- Factor 8 : Converts data size from bytes to bits

- **Path Loss:** Aggregated using both the mean and the variance. The mean captures the average propagation conditions, while the variance reflects the spread of UE conditions within the cell.
- **UE SRS SINR:** Aggregated as the mean and the variance of SRS SINR across all UEs, providing an estimate of the overall uplink channel quality in the cell.
- **Layers:** Aggregated with the mean of how many layers on cell level for a UE using SU-MIMO reciprocity.
- **Pre-estimated UE SRS SINR:** This feature is the estimated SRS SINR before a UE is allocated with an SRS resource. This is aggregated using both the mean and the variance.
- **Downlink SRS SINR:** This feature is the actual downlink SRS SINR for both reciprocity and codebook UEs.

  This is aggregated using both the mean and the variance.
- **Total Downlink BLER:** This is the total downlink Block Error Rate. Aggregated using both the mean and the variance.
- **Aggregated Throughput:** Computed as the throughput across all UEs, representing the overall data rate performance of the cell. The aggregated throughput is approximated as follows:

$$\bar{T} = \sum_{i=0}^{n} (f_i \cdot S_i) \cdot 8 \cdot 10^{-6}, \text{Mbit/500ms}$$

  - $\bar{T}$: Aggregated throughput over the aggregation period (Mbit/500ms)
  - f_i : Indicator variable (new data flag), equals 1 if new data is transmitted, otherwise 0
  - S_i : Size of transmitted data in bytes
  - $\sum$: Summation over all observations i within the aggregation period
  - Factor 8 : Converts bytes to bits
  - Factor 10^{-6} : Converts bits to megabits (Mbit)

- **Interference-plus-Noise (IpN):** IpN represents the total unwanted signal power received, including both interference from other transmitters and background noise, and is averaged on cell level [2].

3.5 Characteristics of the Prepared Data

The final dataset consisted of around 2300 data points. A clear characteristic of the data is the presence of many outliers and extreme values. This is mainly because the data was collected from a real world network, where a wide range of conditions can occur. For example, UEs may experience different signal qualities, changes in network load, movement between cells, or short interruptions in connectivity. These factors can result in unusually high, low, or sometimes missing values. As a result, the dataset reflects realistic network behavior, including noise and irregularities. These characteristics make the data more challenging to model, since outliers can affect the performance of some ML methods. At the same time, it provides a more realistic basis for evaluating how well the models generalize in practice.

Chapter 4
Analysis

This section describes how the data was analyzed in this thesis. To ensure that the data were accurate and reliable, it needed to be analyzed prior to being used in the ML model.

4.1 Analysis of Initial Approach

As previously mentioned when discussing the initial approach, in Chapter 3, the first data sets ordered from the real live network used the lowest SRS SINR threshold (referred to as Threshold A). The purpose of using a low SRS SINR threshold was to allow many UEs to be granted SRS, in order to obtain a large amount of data for analysis.

4.1.1 Feature Enabled

A specific feature was enabled that restricted the use of reciprocity. This resulted in only UEs with very good radio conditions (typically those located close to the base station) being selected to use reciprocity-based beamforming, the others falling back to using codebook-based beamforming.

Figure 4.1 shows SRS SINR on the x-axis and SE on the y-axis. UEs using codebook and reciprocity were visualized separately in order to compare the performance of the two precoding methods. When generating this figure, UEs that were not granted SRS were excluded from the comparison since in the initial approach, only the UEs using SRS was of interest. The background bars in Figure 4.1 indicate the number of samples corresponding to each SRS SINR bin for a given precoder. The solid line illustrates how SE varies with SRS SINR. For each SRS SINR bin, the circular markers represent the average SE for the respective precoder, while the vertical lines indicate the standard deviation around that mean. Longer vertical lines, i.e. higher standard deviation, correspond to higher variability. As seen in Figure 4.1, these vertical lines were present. The observed variability may be attributed to several factors, such as UE mobility and other changing channel conditions.

As stated earlier, the hypothesis was that there would be a crossing point between reciprocity and codebook UEs. More specifically, it was expected that codebook would perform better at lower SRS SINR values. Above Threshold A,

at some specific SRS SINR value, a crossing point was expected where reciprocity would start to achieve higher SE values on average than codebook for the same SRS SINR values. This would be the optimal threshold for that specific radio condition. Nevertheless, this was not observed in the data, see Figure 4.1. Instead, no such crossing point could be identified. UEs with reciprocity achieved higher spectral efficiency on average than the UEs using codebook at all times. A reason as to why this occured, could be due to the feature being enabled, restricting the UE to be of very good radio conditions in order to use reciprocity. The UEs with reciprocity, are of such good radio conditions, that they will always perform better than the UEs with codebook.

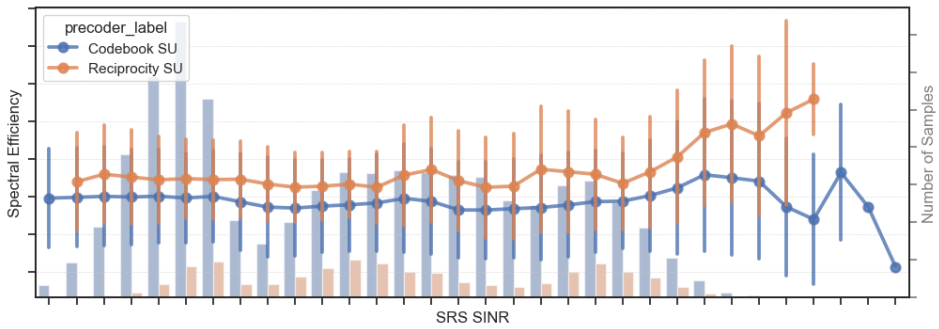

Figure 4.1: Visualization of the first dataset with the feature enabled. Mean Spectral Efficiency is shown on the y-axis and SRS SINR on the x-axis.

The analysis continued by visualizing the toggling modes of the data. As previously mentioned in Chapter 3, the data that was collected was toggled between reciprocity and codebook. For this, Figure 4.2 was generated. This figure shows how often a specific parameter, the one that says which precoder type is used, is set to reciprocity, over time. The data is divided into time intervals of 15 seconds, in total of six minutes.

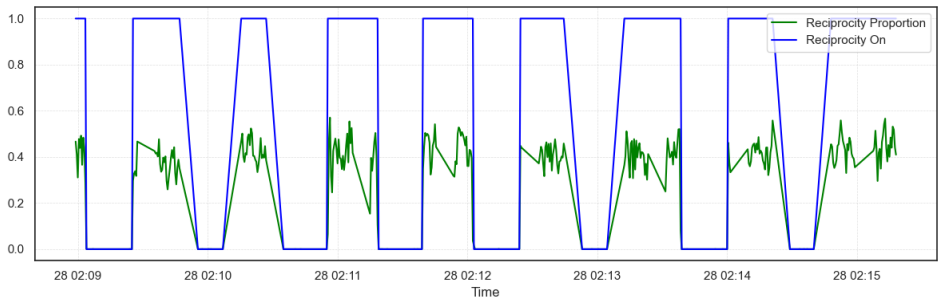

Figure 4.2: Visualization of the first dataset with reciprocity proportion and reciprocity on, with the feature enabled for threshold A.

For each interval, the following is calculated:

- **Reciprocity Proportion:** The amount of data points in the interval where the precoder type is set to Reciprocity. This is done by calculating the mean of how many UEs that has reciprocity-based beamforming at that specific time. This is the green line.
- **Reciprocity On:** If the mean of Reciprocity Proportion is at least at a certain value, that means Reciprocity is on. Otherwise, it is set to zero, which means Reciprocity is off. This is marked in the Figure as the blue blocks.

This figure (Figure 4.2) was created for two reasons. Firstly, it serves to verify that the data collection instructions were followed correctly. Specifically, reciprocity was intended to be active in 15 second windows over a period of six minutes, which should result in eight blue blocks. Secondly, the hypothesis was that when reciprocity was enabled, close to 100% of the UEs would use reciprocity. However, this was not observed. When the feature was enabled, the result was 40%, which can be seen in Figure 4.2. To better understand this outcome, experiments were conducted involving further filtering of the parameters. For instance, filtering for bandwidth equal to 100 MHz and removing carrier aggregation, as previously mentioned in Chapter 3, was applied.

4.1.2 Feature Disabled and Further Filters Applied

Together with the further filtering, the feature was then now also disabled. However, when the feature was disabled, the data collection did not include Threshold A due to data collection execution error. Nevertheless, Threshold B could be analyzed as the second lowest threshold, following Threshold A, and the resulting Reciprocity Proportion reached around 60% as seen in Figure 4.3 This was higher than before but not reaching 100% as anticipated.

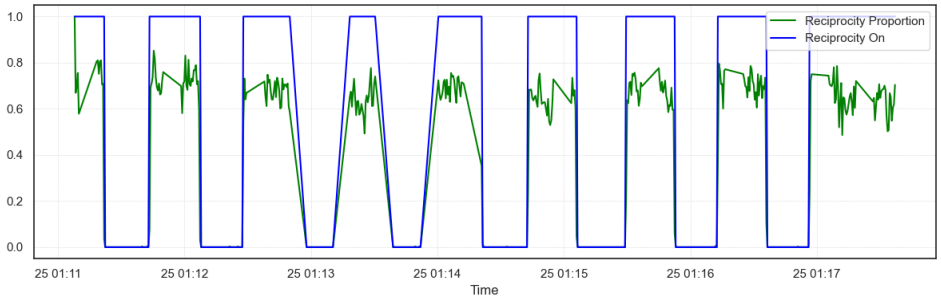

Figure 4.3: Visualization of a dataset with reciprocity proportion and reciprocity on, with the feature disabled for threshold B.

After collecting the data set with the feature disabled, the hypothesis regarding the crossing point was revised. When observing Figure 4.4, multiple crossing points could be observed instead of only one. This meant that the hypothesis was not

correct, no specific optimal, unique threshold existed. A detailed analysis of the underlying reasons for this behavior is beyond the scope of this thesis, but the main assumption was not seen. As a result, the analysis shifted focus towards the other approach.

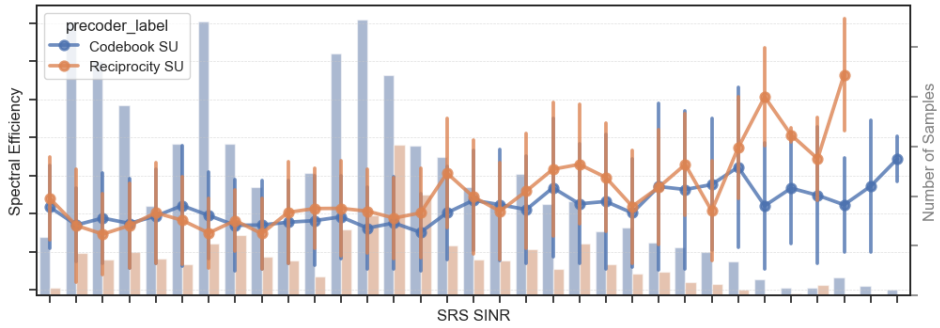

Figure 4.4: Visualization of a dataset with the feature disabled. Mean Spectral Efficiency is shown on the y-axis and SRS SINR on the x-axis.

4.2 Analysis of the Evolved Approach

To ensure the quality and reliability of the data set used in the evolved approach, several validation steps were performed. Firstly, it was verified that the reverse order had been applied correctly. This was done by examining which thresholds were present in each log and their respective proportions, see Table 4.1. As can be seen in Table 4.1, the thresholds appear in each log in ascending order, Log 1 containing only Threshold E and C, and so on. This confirms that the setting with reverse order was applied. For Log 1 and Log 2, the intended threshold had the second-highest proportion, after Threshold C. This indicates that Threshold C partially overwrites the intended thresholds in these logs. For Log 3 and Log 4, however, the intended thresholds (i.e., Threshold B and Threshold A, respectively) had the highest proportion, which is desirable as it results in a larger amount of data for the intended thresholds.

Log 1	
Threshold	Proportion
E	0.330768
C	0.669232

Log 2	
Threshold	Proportion
E	0.084232
D	0.406810
C	0.508958

Log 3	
Threshold	Proportion
E	0.056926
D	0.072096
C	0.359920
B	0.511058

Log 4	
Threshold	Proportion
E	0.216830
D	0.077508
C	0.326505
B	0.154664
A	0.419641

Table 4.1: Proportional distribution of thresholds per log.

This was important to confirm that the experimental setup followed the intended structure. Second, the proportion of samples corresponding to each threshold was examined, ensuring that a high percentage was present for each threshold. However, these validations alone were not sufficient to fully assess the quality of the data. Therefore, additional analysis was conducted. Specifically, the number of unique UEs in each SRS log was evaluated. Furthermore, for each log, the number of UEs whose SRS SINR fell below the intended threshold was analyzed. The results showed more UEs with SRS than expected. A lower amount of UEs with SRS was expected since for the second and evolved approach, both SRS and non SRS users were allowed, to minimize bias towards higher thresholds (as explained in Chapter 3).

Since the data was collected in reversed order, it was important to verify that UEs were not retaining lower thresholds that could overwrite or interfere with the intended ones. The results of this can be seen in Table 4.2. As mentioned, Threshold C was used as the fixed baseline threshold. UEs configured with Threshold C could already have an active connection to the base station before the data collection started, that unintentionally overwrote the collection of higher thresholds; Threshold D and Threshold E. This could explain why these thresholds contained 27 and 31 UEs, respectively, see Table 4.2, with SRS SINR values below the intended threshold range. Despite this, the number of such UEs remained relatively small, indicating that this issue did not significantly affect the data.

Log	Total number of unique UEs	Number of unique UEs with SRS	Number of UEs that have SRS SINR below the intended threshold	Intended threshold
Log 1	271	236	27	E
Log 2	306	257	31	D

Table 4.2: Summary of log values for the final dataset.

Chapter 5
Machine Learning Model

This part of the thesis presents the reasoning and choice of the ML models evaluated and usef. It also includes the metrics used for testing the model and how it was validated.

5.1 Model of Choice

The analysis provided by Achshah and Prakash in [14], offers an overview of various ML architectures and visual data to highlight the respective strengths and limitations of each model presented. This framework serves as a guide for selecting an architecture suited to the specific constraints of this research. Since the primary goal is to predict a continuous target variable, the SE, regression based models are the most appropriate choice [14]. However, the relationship between the input parameters and the target variable is expected to be complex and non-linear. The reason for this is that the data used for this thesis originates from a real live network where the conditions are always changing. This limits the effectiveness of simpler models such as linear regression [14].

5.1.1 Random Forest Regressor

In Skäremos thesis [3], the conclusion was that of all the models tested, Random Forest performed the best with an accuracy of 75.47%. In Skäremos thesis, the model was used to determine, based on different parameters, when to use SRS and when to avoid it. This supports Random Forest's suitability for similar problem settings, as in this thesis.

Random Forest is an ensemble method that builds multiple decision trees during training and outputs the mean prediction of the individual trees. The different trees are trained on partially overlapping subsets of the data, which helps diversify the model. This technique is called bagging. One of its main strengths is its ability to reduce overfitting by averaging across multiple models. Additionally, and important to this thesis, the method is robust to outliers, leading to a more stable model with better generalization abilities [15]. Another advantage of Random Forest is its performance on relatively small datasets. Small datasets are particularly vulnerable to overfitting since complex models easily can memorize the few available examples. However, since each tree is trained on a random subset of both

the data and the features, the model can generalize well even when the number of observations are limited [16]. Given the limited amount of data and the high variability present in this study, these properties make Random Forest a strong baseline model.

5.1.2 Extreme Gradient Boosting (XGBoost)

Although suitable for this thesis, the Random Forest model showed signs of overfitting, which is described in more detail later in this chapter. To further investigate whether model performance could be improved, Extreme Gradient Boosting (XGBoost) was explored.

XGBoost is an ensemble method based on sequential learning (a technique known as boosting), where each model focuses on correcting previous errors [17]. This allows the model to capture complex data dependencies. XGBoost is particularly known for its ability to reduce overfitting [17]. In order to do this, XGBoost employs a regularized learning objective that penalizes model complexity, applies shrinkage to scale down newly added weights, and utilizes column subsampling (meaning it only considers a subset of features when building each tree, similar to Random Forest models).

By experimenting with both Random Forest and XGBoost, the study aims to evaluate whether a boosting based approach can provide improved generalization and predictive performance compared to the bagging based baseline.

5.1.3 Machine Learning Framework

The machine learning models were implemented using two different frameworks, scikit-learn and the XGBoost library. Scikit-learn is an open source machine learning library written in Python [18], while XGBoost is a gradient boosting framework developed by the Distributed Machine Learning Community (DMLC) [19]. From scikit-learn, the **RandomForestRegressor** [20] was used to implement the Random Forest model. From the XGBoost library, the **XGBRegressor** was used to implement the XGBoost model. Both implementations provide a scikit-learn compatible interface for training and evaluation, enabling a consistent workflow across models.

For both models, an initial evaluation was performed using the default hyperparameter settings provided by their respective libraries. This was done to establish a baseline performance and to serve as a reference point for determining whether further hyperparameter tuning was necessary. The same evaluation procedure and experimental steps were applied to both models to ensure a fair and consistent comparison. An optimization of the hyperparameters was performed to ensure that the most suitable parameters was used. For this, the **RandomizedSearchCV** [21] was used, also from the library scikit-learn. This method evaluates randomly chosen combinations of parameters through cross-validation to identify the most effective configuration. The hyperparameters for Random Forest can be seen in Table 5.1, and the hyperparameters for XGBoost can be seen in Table 5.2.

To implement the method **RandomizedSearchCV**, a search interval had to be established. This was done by including values both under and over the refer-

ence points of the standard settings. Compared to the method **GridSearchCV** [22] which tests all possible combinations and returns the best from these, the **RandomizedSearchCV** tests only a selection. This method is more time efficient than testing every possible combination.

Parameter	**Values**
n_estimators	100, 200, 300
max_depth	5, 10, 15, 20
min_samples_split	10, 15, 20
min_samples_leaf	1, 2, 5, 10
max_features	sqrt, log2, 0.5

Table 5.1: Hyperparameter range for the Random Forest model.

Parameter	**Values**
n_estimators	100, 200, 300, 400
max_depth	2, 3
learning_rate	0.01, 0.05
subsample	0.7, 0.8
colsample_bytree	0.7, 0.8
gamma	0, 0.1, 0.5, 1, 5
reg_lambda	1, 5, 10
min_child_weight	5, 10

Table 5.2: Hyperparameter range for the XGBoost model.

5.2 Features and Target Values

The dataset used was divided into two parts:

- Target Value Y: The value to predict. This is the Spectral Efficiency.
- The Features X: The inputs used to make the prediction. For this, all the data was used except for the columns containing the SE, the id of the cell, the name of the log and the time. These columns should not be used as inputs for training the model.

To test the model's performance, the data was split into a training set (80%) to teach the model and a test set (20%) to evaluate how well it works on new data.

5.3 Model Validation

To ensure reliable generalization, as previously mentioned, the model was evaluated using an 80/20 train-test split. K-fold cross-validation was used on the training set, and final performance was assessed on a separate test set.

5.3.1 Metrics

To evaluate the model's predictive performance, three standard regression metrics were employed: Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Error (MAE). MSE measures the average of the squares of the errors, meaning the average squared difference between the estimated values and the actual values. It is defined as follows [23]:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2,$$

where n is the amount of data points, y_i is the actual target value and $\hat{f}(x_i))$ is the prediction that $\hat{f}$ gives. If the prediction output values and the actual output values substantially differ, the MSE value will increase [23]. The goal is to have it as low as possible both on the training data and on the test data. RMSE is another common metric that takes the square root of the MSE value, making it easier to interpret, since it has the same unit as the target variable. It is defined as follows [24]:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2},$$

where n is the amount of data points, y_i is the actual target value and $\hat{f}(x_i))$ is the prediction that $\hat{f}$ gives.

MSE and RMSE are both sensitive to outliers because they square the errors, making large mistakes have a greater effect on the result. MAE, on the other hand, treats all errors linearly and gives a better picture of the average performance [24]. By comparing both, there is a possibility to see if the model's mistakes are mostly small and steady, or if they are caused by a few very large errors. To understand the average error rate in a model without it specifying if it is under or over performing, MAE can be used. When calculating MAE, the errors are not squared which means that MAE treats all errors with equal importance, the formula is defined as follows [25]:

$$MAE = \frac{\sum_{i=1}^{n}\left|y_i - \hat{f}(x_i)\right|}{n},$$

where n is the amount of data points, y_i is the actual target value for a specific data point and $\hat{f}(x_i))$ is the predicted value for this data point. MAE is, when compared to the other previous mentioned metrics, not as sensitive to outliers. Another advantage of the metric MAE is that it is in the same unit as the target variable, which makes it easier to comprehend. The same advantage applies to RMSE. The lower the MAE value, the higher the precision of the predictions [25].

5.4 Training the Model

Initial results with Random Forest showed promising results, see Figure 5.1. Figure 5.1 was visualised with scikit-learn's **PredictionErrorDisplay** [26]. In this figure, the diagonal line represents a perfect prediction where the estimate equals the actual value. Any deviations from this line indicate prediction errors. Given the natural noise in the dataset, a perfect fit is often not possible. A reliable model is identified by how closely its predictions cluster around the diagonal, showing that the errors are small and consistent [26].

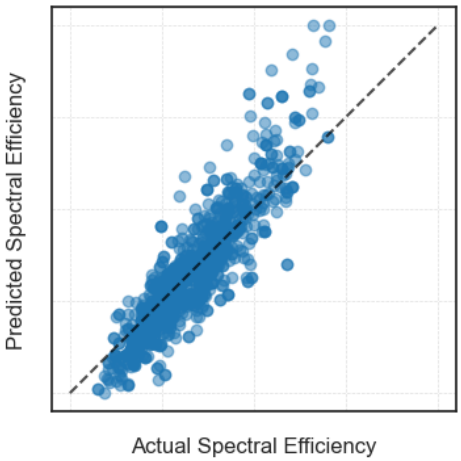

Figure 5.1: A PredictionErrorDisplay plot with the training data for the Random Forest model. The actual values is shown on the x-axis and the predicted values are on the y-axis.

In Figure 5.1, most of the data points are close to the reference line. There are however some data points that appear far from the line, a few outliers, but most of them are centered in the middle.

5.4.1 Overfitting

On the other hand, when analyzing the metrics for the Random Forest model, the model achieved a low training error according to the RMSE value but the error nearly doubled when evaluated on the test set, indicating overfitting. This suggests that the model has been overly adjusted to the training set and therefore struggles to generalize effectively to unseen data [27]. The cause of this could primarily be the excessive complexity of the model, where deep trees allowed the model to capture noise rather than general patterns. To address this, the complexity of the model was constrained by limiting the depth of the tree. This resulted in a more balanced performance, where the training error and test error were much closer. Although MAE increased slightly, the proximity between these values indicates improved generalization and a more robust model. This transition reflects the

bias-variance tradeoff, where reducing the complexity of the model lowers the variance at the cost of increasing the bias [28].

To improve performance further considering the overfitting, XGBoost was evaluated as an alternative method. The RMSE values with XGBoost still indicated a degree of overfitting, as the error remained higher on the test set than on the training set. To address this, feature selection could be implemented as a further refinement. Feature selection means reducing features that do not seem to affect the target value, these unimportant features can be seen as noise, meaning data that confuses the model [29]. The importance of the different features in X can be seen in Figure 5.2.

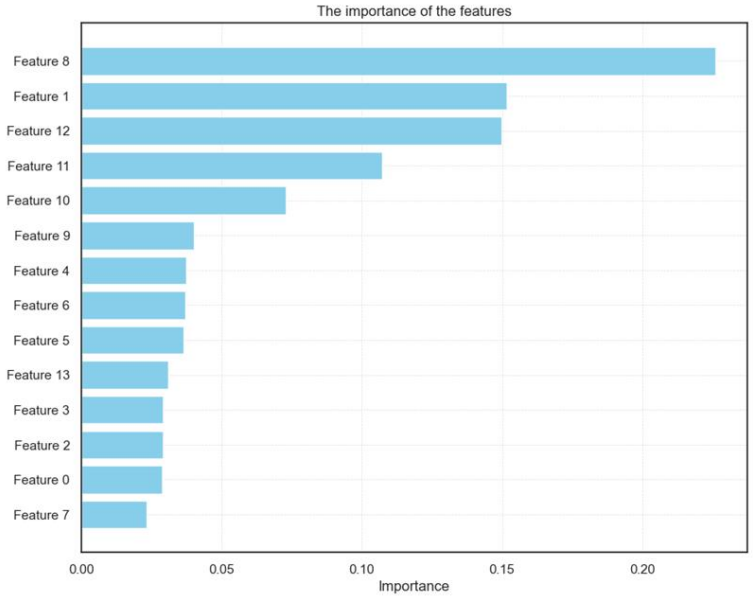

Figure 5.2: Visualization of feature importance from feature selection. The feature names have been anonymized.

In Figure 5.2, all features have a high level of importance and thus no feature should be excluded from the set. Since the degree of overfitting was still reduced, i.e the RMSE value of the test set decreased, compared to the Random Forest model, this was considered acceptable.

To further understand how the important features impact the model's output, a SHAP figure, using the library shap [30] was used, which can be seen in Figure 5.3. The plot in the Figure 5.3 is called a Beeswarm plot and provides a compact overview of how the most important features in the dataset influence the model predictions of the target value [30].

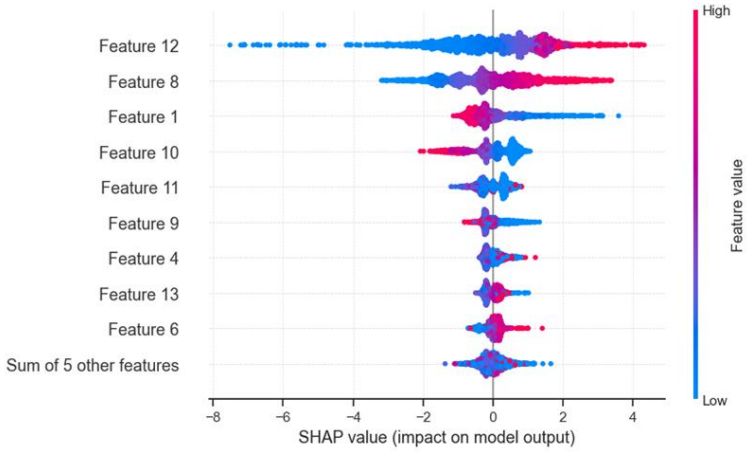

Figure 5.3: A Beeswarm SHAP figure.

The figure shows how the features affect the models predictions. A dot in Figure 5.3 represents one data point. Blue points are low values and pink/red values represent high values. Data points which are to the right of the middle line increases the prediction of the target value. Data points which are to the left of the middle on the other hand decreases the prediction of the target value [30]. Examining the Figure 5.3, shows that low values of the aggregated throughput decreases the predicted target value. The mean of the SRS SINR for a UE, shows that high values of this feature increases the predicted target value.

A similar PredictionErrorDisplay figure was conducted for the XGBoost model, as shown in Figure 5.4. This figure shows that the model performs slightly better than before since it has less deviations than the Figure 5.1.

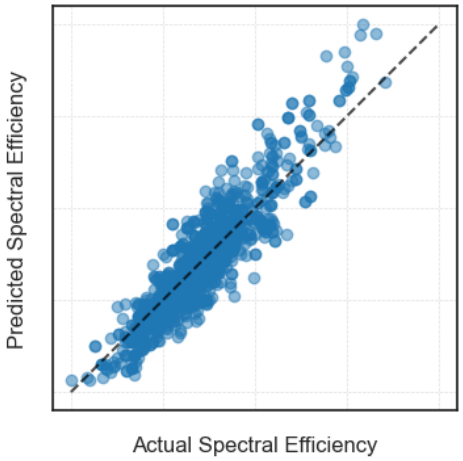

Figure 5.4: A PredictionErrorDisplay plot with the training data for the XGBoost model. The actual values is shown on the x-axis and the predicted values are on the y-axis.

As a result, XGBoost was adopted as the primary ML model for the remainder of this study.

5.4.2 Extreme Values

During the model evaluation, the model struggled to accurately predict extreme values. To investigate this further, a Normal Q-Q plot was generated to determine if the residuals followed a normal distribution, see Figure 5.5. A Normal Q-Q (Quantile-Quantile) figure is used to determine whether residuals follow a normal distribution. When the data points are aligned with the straight diagonal line, the residuals are considered to be normally distributed. Any significant deviation from this line suggests that the data does not follow a normal distribution [31]. This means that the data contains unevenly distributed values, which may include extreme values.

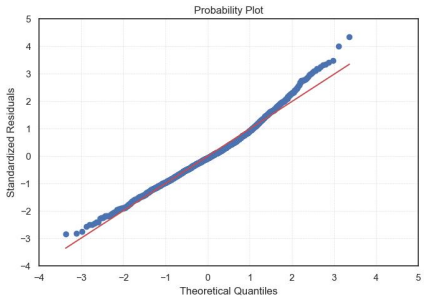

Figure 5.5: A Normal Q-Q plot with the XGBoost model.

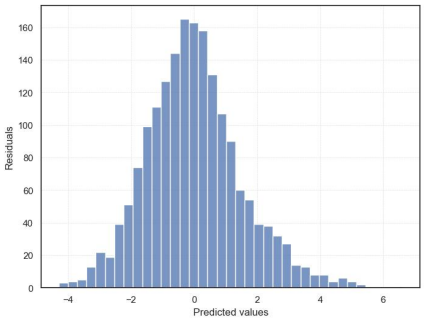

Figure 5.6: A Residuals Histogram with the XGBoost model.

In Figure 5.5, the data points show a noticeable deviation from the reference line. This is further supported by Figure 5.6, where the distribution appears to be right-skewed. This irregularity correlates with the observations in the Normal Q-Q plot, which displays a "heavy tail" at the higher quantiles. Such deviation indicates that the residuals are not normally distributed, specifically, the presence of these heavy tails suggests an excess of extreme positive and negative residuals compared to what would be expected under a normal distribution [32]. This heavier tail that was observed for the final model might be due to the fact that higher SE values are generally less common in the network compared to more average SE values. This results in fewer training samples representing these conditions. Consequently, the model has limited exposure to scenarios with high SE during training, which reduces its ability to accurately predict these values compared to the more frequently occurring average SE values.

For comparison, these are the same plots, the Normal Q-Q plot and the residuals with Random Forest:

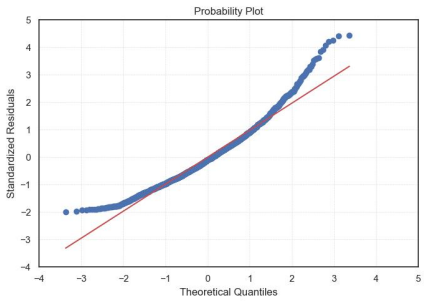

Figure 5.7: A Normal Q-Q plot with the Random Forest model.

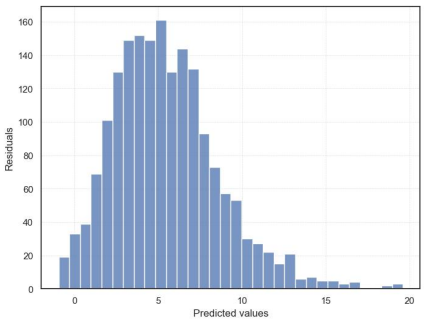

Figure 5.8: A Residuals Histogram with the Random Forest model.

In Figure 5.7 and Figure 5.8, it can be seen that the distribution is even more

positively skewed and has an even heavier tail at the higher quantiles, which further supports the choice of choosing the XGBoost model. As previously mentioned and as seen in Figure 5.4, the XGBoost model shows an improved ability in predicting extreme values compared to Random Forest in Figure 5.1. Because of this, as mentioned earlier, this model will be used for the remainder of the study.

Chapter 6
Results and Discussion

This section presents the results of this thesis and associated discussions. This section also addresses the limitations of the findings.

6.1 Limitations of the Thesis

Obtaining reliable and representative results in machine learning is challenging, particularly in wireless network environments where conditions continuously vary over time. Several factors affected both the quality of the collected dataset and the validation of the developed model. These limitations are discussed in the following sections.

6.1.1 Date and Time of Data Collection

One important challenge was that the data was collected at different points in time. Since the network environment of a real world network is constantly changing, the radio conditions, UE distribution, traffic load, and radio frequency characteristics were not identical between measurements. Consequently, it was not possible to guarantee exactly the same conditions for the datasets corresponding to each threshold configuration. To reduce this effect, the datasets were collected consecutively during the same date and time period. However, the ideal scenario would have been to collect all threshold datasets simultaneously under identical network conditions, but this is not possible in practice.

6.1.2 Limited Data Availability

Another limitation was the restricted amount of available data. Due to difficulties in collecting threshold logs in live networks, the only dataset containing the reverse order scenario originated from a single day. This may introduce bias, since ML models generally benefit from training on data collected during multiple days and under diverse conditions. Such diversity improves the model's ability to generalize to unseen scenarios. Despite this limitation, the final dataset consisted of approximately 2300 data points, which was considered sufficient for the scope of this thesis.

6.1.3 Overlapping Thresholds

A further challenge, and likely the most significant limitation, was the overlap between thresholds. In every collected log, Threshold C partially overwrote the intended threshold during data collection. The UEs associated with Threshold C were typically those maintaining strong radio connections and therefore operating under very favorable radio conditions.

As previously mentioned in Chapter 3, since Threshold C did not have its own log, it was collected from Log 1 (belonging to Threshold E originally). In this dataset, as seen in Table 4.1, Threshold C represented approximately 70%, a higher number than for Threshold E. In other words, Threshold C data originated from scenarios where it overrode Threshold E. This imbalance likely introduced bias into the model training process. Since the model was exposed predominantly to UEs with strong radio conditions, it became biased toward predicting Threshold C more frequently than other thresholds. This behavior can also be observed in Table 6.3, where the model showed a tendency to favor this threshold during prediction. Additionally, the overlap between thresholds resulted in less clean data overall. None of the collected logs contained data corresponding exclusively to a single threshold. As a consequence, the effective amount of usable training data for each threshold was reduced, limiting the amount of representative information available during model training.

6.1.4 General Limitations of Validating Machine Learning Models

Finally, validating the proposed solution in a real world deployment proved to be highly challenging. Integrating such a solution into daily network operation is difficult, and determining whether the model selects the optimal threshold in a previously unseen scenario is not straightforward. During training, the correct labels are known. However, in new situations, the optimal threshold which results in the highest SE is unavailable. Consequently, validation must rely on indirect evaluation methods and assumptions regarding the model's generalization capability. For this reason, an alternative validation strategy was adopted in this thesis. A more detailed description of this evaluation methodology is presented in Section 6.2.

6.2 Result of The Machine Learning Model

The model was analyzed in terms of how well it performs at predicting an optimal threshold, i.e., a threshold that results in the highest SE. This was done by comparing the actual threshold used in a specific context with the threshold predicted by the model, and evaluating the SE obtained in each case. Here, the "actual threshold" refers to the threshold that was applied in the given scenario (one of the discrete options A–E), and not necessarily the optimal threshold. For each scenario, the SE achieved using the actual threshold was first determined, after which it was compared to the SE obtained using the threshold selected by the ML model. In this way, the analysis evaluates whether the model is capable

of predicting a threshold that results in higher SE than the one statically set and originally used.

The analysis distinguishes between cases where the model predicts the same threshold as the actual one and cases where it predicts a different threshold. When the predicted threshold differs, the resulting SE values are compared to determine whether the models choice leads to a higher or lower SE than the actual threshold. In this way, it is possible to evaluate how often the model suggests a threshold that improves or degrades SE compared to the one that was originally used. The following results were obtained, see Table 6.1:

Table 6.1: Results of the ML model when predicting an optimal threshold.

Scenario	Number of scenarios	Share (%)	Average SE difference (bits/Resource Elements)	Expected SE difference (bits/Resource Elements)
Model suggests a threshold that leads to higher SE	1108	48.3	1.261801	1.261801
Model suggests a threshold that leads to lower SE	801	34.9	-1.300877	-1.300877
Model suggests keeping the same threshold	387	16.9	-0.367044	0

The results show that in a majority of the scenarios (48.3%), the ML model suggests a threshold that results in higher SE compared to the actual threshold, with a positive average SE difference of 1.26 bits/Resource Element. In contrast, the model suggests a worse threshold in 34.9% of the cases, leading to a negative average SE difference of 1.30 bits/Resource Element. In 16.9% of the scenarios, the model agrees with the actual threshold. Overall, this indicates that the model more frequently identifies thresholds that improve SE than those that degrade it. The expected SE difference can also be observed in Table 6.1. When the model recommends using the same threshold as the static one, the expected SE difference should ideally be 0. However, this is not the case, which may be due to the ML model not being able to predict SE with 100% accuracy.

6.2.1 Net Expected Spectral Efficiency (Net Value)

To fully quantify the overall impact of the dynamic threshold across the network, a net expected Spectral Efficiency (net value) was calculated. By only observing the win and loss ratios (48.3% increase vs. 34.9% decrease), a conclusion about the degree of the respective changes cannot be made. The net value provides the average expected change in SE and is a direct application of the statistical concept of the expected value for a discrete random variable [33]. By using the general definition of expected value to the outcomes, the formula is defined as:

$$Net = (P_{\text{gain}} \times \Delta SE_{\text{gain}}) + (P_{\text{loss}} \times \Delta SE_{\text{loss}})$$

where P_{gain} and P_{loss} represent the proportions of cases with improved or degraded SE, respectively, and ΔSE is the average performance change for gain and loss, respectively. Since cases with unchanged performance yield a ΔSE of zero, they

do not affect the net calculation. This means that when the model keeps the same threshold as the static one, the SE will not change and is therefore not included in the equation. By applying this expression, the resulting net value was calculated and the result were:

$$\textbf{Net value} = 0.145139 \approx 0.15 \text{ bps/Hz}.$$

This positive net value indicates that, although the model reduces performance in around one third of the cases, the overall effect of the dynamic threshold is still an increase in spectral efficiency across the network.

6.2.2 Histograms of Model Performance

To further visualize the models performance, histograms were generated, as shown in Figure 6.1. These histograms illustrate the different scenarios. The top image represents the case where the ML model selects a threshold that results in a higher SE compared to the actual threshold. The middle image corresponds to the scenario where the ML model predicts the same threshold as the actual one. As shown in Figure 6.1, both the model and the actual threshold follow a similar overall distribution, although the model exhibits a generally higher density. This higher density indicates that the model tends to more frequently predict SE values within this distribution, suggesting a bias or preference toward specific regions of the SE space. The bottom image illustrates the scenario where the ML model proposes a different threshold than the actual one, and where this choice results in a lower SE.

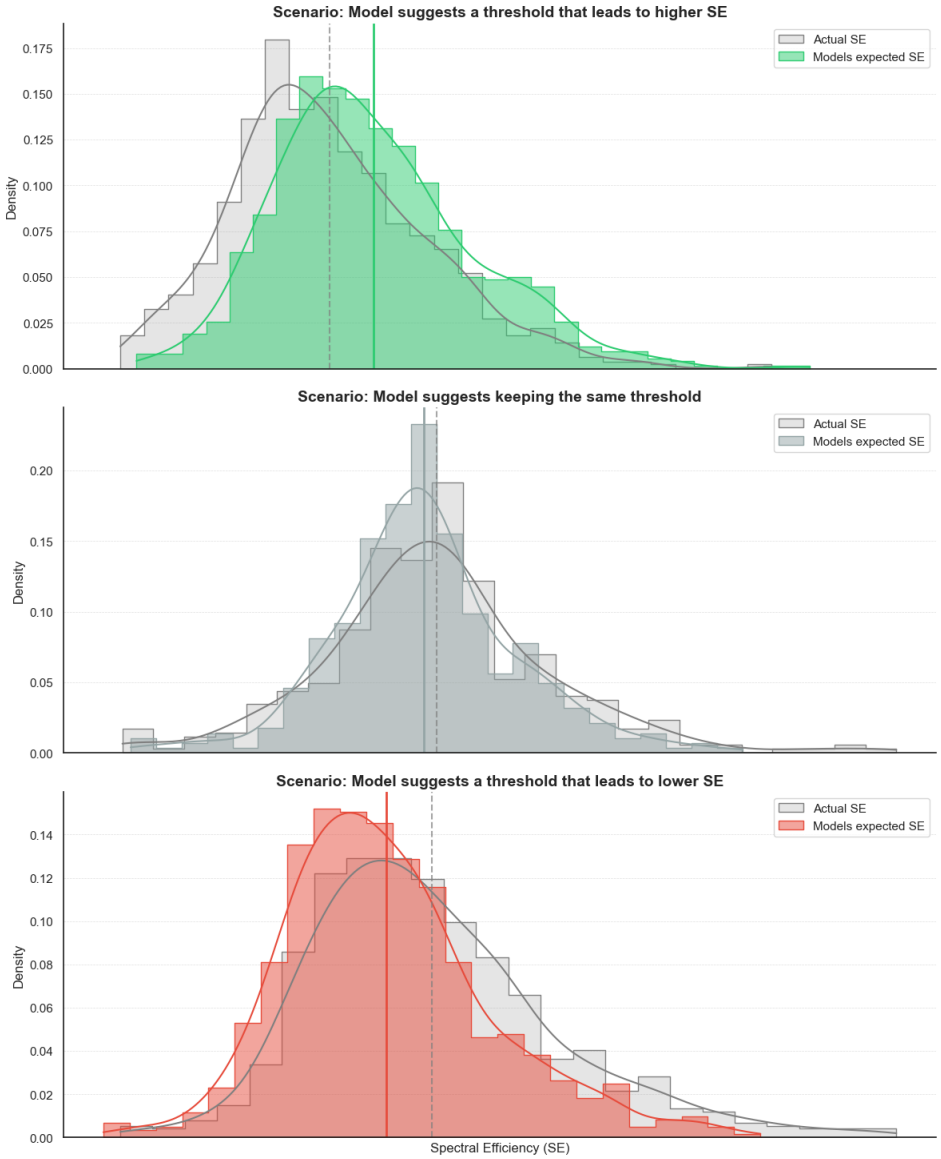

Figure 6.1: Histograms of the three scenarios. Density can be shown on the y-axis and SE on the x-axis.

6.2.3 The Models Predictions

In Table 6.2 and Table 6.3, the distribution of the thresholds and their corresponding proportions can be observed.

Table 6.2: The actual threshold distribution

Threshold	Proportion
A	0.214286
B	0.242596
C	0.192073
D	0.195557
E	0.155488

Table 6.3: The models threshold prediction distribution

Threshold	Proportion
A	0.187718
B	0.025697
C	0.679443
D	0.098432
E	0.008711

Table Interpretation

Table 6.2 presents the actual thresholds that were statically applied during testing and therefore acts as the reference distribution against which the model predictions are compared. For example, Threshold B was the applied threshold in approximately 24% of the evaluated cases, see Table 6.2. This means that, in 24% of the test samples, the SE obtained from using Threshold B statically was considered the reference outcome.

The purpose of comparing this distribution with the model predictions is to analyze whether the model tends to favor different thresholds. For instance, if Threshold B appears in 24% of the actual cases but is rarely or never predicted by the model, this may indicate that the model considers other thresholds to provide better expected SE under those conditions. As shown in Table 6.3, the model predicted Threshold C in approximately 68% of the evaluated cases, representing a substantial majority of the predictions. This suggests that the model frequently estimated Threshold C to provide higher SE compared to the statically configured thresholds.

Discussion of the Tables

It is important to note that the model predicted a higher SE than the static configuration in approximately 48% of the cases and not 100%. Therefore, the predictions should not be interpreted as absolute ground truth, but rather as the model's estimated optimal choice under the given conditions. The strong preference for Threshold C is also likely influenced by the dataset limitations discussed previously. In particular, the imbalance caused by the overlap between thresholds resulted in Threshold C dominating a large portion of the training data, which may have biased the model toward favoring this threshold during prediction.

6.3 Results and Discussion of RQ1: Design of the Machine Learning Model to Dynamically Set the Threshold

The findings showed that the XGBoost model outperformed the Random Forest model in terms of overall predictive performance. As illustrated in the corresponding result figures, see Chapter 5, XGBoost demonstrated a better ability to capture the relationships within the dataset and produced more accurate predictions across most scenarios. However, despite the improved performance, the model still struggled with extreme values and certain outlier conditions, indicating that further optimization and additional data could potentially improve the results. More about the further improvements can be read about in Section 7.3.

A significant finding throughout this thesis was that the quality and structure of the data had a major impact on the model performance. A large portion of the work therefore focused on preprocessing, filtering, and aggregating the collected network data in order to construct a suitable dataset for machine learning. When the data was noisy, inconsistent, or insufficiently processed, the model performance weakened considerably, highlighting the importance of reliable data preparation.

Another important aspect of the model design was the effort to create a more generalized model rather than a highly specialized one. Preventing overfitting was a key consideration during development, since the objective was not only to achieve good performance on the training data, but also to ensure that the model could generalize to previously unseen network conditions. For this reason, the selected features, aggregation strategies, and model parameters were chosen with the aim of improving robustness and reducing the risk of overfitting to specific measurement scenarios.

6.4 Results and Discussion of RQ2: Input Parameters with the Highest Impact

To better understand how the different input parameters influenced the model predictions, both feature importance analysis and SHAP values were examined. These methods provide insight into how the selected features contributed to the prediction of the target value, namely the predicted SE, which was then indirectly used to determine the optimal threshold.

6.4.1 SHAP Results

By examining Figure 5.3, it can be observed that the aggregated throughput had a significant influence on the model output. Lower values of this feature generally pushed the prediction toward lower target values, while higher throughput values contributed positively to the predicted SE. The mean SRS SINR for a UE, showed a strong positive relationship with the prediction. Higher values of the mean SRS SINR tend to increase the predicted target value, indicating that favorable radio conditions were associated with higher predicted SE. The SHAP analysis also demonstrated that the features affected the model output in different ways. Both the aggregated throughput and the mean SRS SINR for a UE followed a

consistent trend where higher feature values (red points) were mainly associated with positive SHAP values, while lower feature values (blue points) were associated with negative SHAP values. This indicates that increasing values of the feature, the mean SRS SINR for a UE and the aggregated throughput, generally increased the predicted target value (SE).

Other features showed a more ambiguous distribution of SHAP values, where both low and high feature values contributed positively and negatively depending on the context (see Feature 5 in Figure 5.3). This suggests that the impact of these features was more dependent on the surrounding network conditions and interactions with other input parameters rather than following a clear trend. The varying distributions observed in the SHAP figure therefore highlight the complexity of the relationship between the network measurements and the resulting threshold prediction.

6.4.2 Feature Importance

These observations are further supported by the feature importance analysis shown in Figure 5.2. The figure presents the relative importance of the features in the trained XGBoost model. In Figure 5.2, the mean of UE SRS SINR, the mean of the pathloss and aggregated throughput were among the most influential parameters in predicting the target value. Several additional features also contributed to the prediction process, although with lower relative importance.

The high importance of path loss is expected. The larger the distance is between receiver and transmitter, the less bites of information can be transferred in a robust way. This aligns with the observed negative relationship between path loss and SE, where higher path loss values correspond to lower SE. The SHAP analysis further supports this relationship, indicating that increased path loss consistently contributes negatively to the models predictions of SE.

It is important to note that the feature importance values reflect the importance of the features when predicting SE rather than directly predicting the threshold itself. Since the threshold selection was based on the predicted SE, the feature importance should therefore be interpreted as the influence of the input parameters on the model's SE prediction, which indirectly determined the chosen threshold.

Another important finding was that the combination of multiple aggregated features enabled the model to capture different aspects of the network state simultaneously. Features related to throughput, radio conditions, and UE behavior collectively influenced the predictions, suggesting that no single parameter alone was sufficient to determine the optimal threshold in all scenarios. This further motivates the use of ML for the problem, since the relationships between the network parameters and the optimal threshold are complex and highly dependent on the current network conditions.

6.4.3 Mean vs Variance Feature Importance

An interesting observation from the feature importance and SHAP analysis is that several features were included in both mean and variance form for the same

metrics. This allows for an analysis of not only the average of a parameter, but also its stability over time. For example, mean of UE SRS SINR, was identified as the most important feature, while SRS SINR variance had significantly lower importance. This suggests that the average radio quality is more influential for predicting the optimal threshold than its short-term fluctuations. In other words, the overall signal quality appears to be more important than how much it varies.

A similar pattern can be observed for uplink pathloss, where the mean of the pathloss ranks high in importance, whereas the variance of the pathloss is among the least important features. This indicates that the mean of pathloss has a stronger impact on the prediction than its variability across time. Overall, these results suggest that the model primarily relies on long-term average network conditions rather than short-term variability when estimating the SE and selecting the optimal threshold.

6.5 Results and Discussion of RQ3: How the Dynamic Threshold Affects the Downlink Performance Compared to Fixed

The results indicate that the proposed approach improves downlink performance, measured in SE, in 48.3% of the cases. It maintains a similar performance in 16.9% of the cases, while resulting in lower SE in 34.9% of the cases, as presented in Table 6.1. All results are computed at cell level. The differences in SE between the dynamic and fixed thresholds, which can be seen in Table 6.1 are relatively small. The average deviation in SE, calculated at cell level, lies approximately within the range of [-1.3, 1.3]. Although the per cell SE improvements appear limited, SE directly relates to overall network capacity. When aggregated across all users within a cell in a real network deployment, even small gains can result in improvements of total throughput and resource utilization.

The net value was calculated to be around 0.15 bps/Hz. This positive result indicates that, although the model predicted a threshold resulting in lower spectral efficiency in approximately one third of the cases, the model's overall performance still outperformed the use of a static threshold.

The main advantage of the dynamic approach is its ability to adapt to local network conditions rather than relying on a fixed configuration. The result of 48.3% suggests that the model is able to identify scenarios at cell level where alternative thresholds yield better SE. However, there is a reason why the dynamic threshold resulted in lower SE in 34.9% of the cases. This is likely due to the data issues mentioned in the previous section, especially the overrepresentation of Threshold C. Since the training data was unbalanced, the model may generalize to heavily. It selects a threshold that performs well on average across the dataset, but which may not be optimal for the specific cells current conditions.

In conclusion, using a dynamic threshold seems to come with a tradeoff. The model is sometimes capable of finding small improvements that fixed thresholds miss, leading to better performance for parts of the network. However, until the model can be trained on a more balanced and diverse dataset, there is still a risk that it will occasionally make bad choices due to bias in the training data.

Chapter 7
Conclusion

In this chapter, the research questions are answered and further discussed. The chapter also presents a summary of the thesis and discusses directions for future work.

7.1 Summary of the Thesis

To summarize, this thesis investigated whether machine learning can be used to dynamically determine the optimal SRS SINR threshold at cell level in a wireless network environment. The study focused on designing and evaluating supervised learning models capable of predicting SE for different threshold configurations based on aggregated network measurements. Two ML models were evaluated, XGBoost and Random Forest, where XGBoost achieved the best overall performance. The results showed that parameters, such as mean UE SRS SINR, mean of the uplink pathloss, and aggregated throughput, had the highest impact on the model predictions on SE. The analysis of the parameters importance also demonstrated that long-term average network conditions were more influential than short-term variability when predicting the optimal threshold.

The proposed dynamic threshold approach improved downlink SE in 48.3% of the evaluated cases compared to fixed thresholds, indicating that adaptive threshold selection has potential to improve network performance. The results also showed a positive net value of around 0.15 bps/Hz. However, the study also identified several limitations related to data quality, threshold overlap, and dataset imbalance, which affected the reliability and generalization capability of the model.

7.2 Research Question Conclusions

The aim of this thesis was to investigate whether a dynamically set threshold would optimize downlink performance, i.e an increased SE, of Massive MIMO on cell level. The research questions, stated in Chapter 1, are addressed in the following sections.

7.2.1 Conclusion of Research Question 1

The results show that an ML model for dynamically determining the optimal SRS SINR threshold at cell level can be designed using aggregated network measurements. Among the evaluated models, XGBoost achieved the best overall performance and demonstrated a stronger ability to capture relationships between the input parameters and the resulting SE. The study also showed that data preprocessing, filtering, and feature aggregation were essential parts of the model design process. The quality and structure of the dataset had a significant impact on the final performance, highlighting the importance of reliable and representative training data when designing machine learning solutions for wireless networks. In addition, the model was designed with a focus on generalization rather than specialization in order to reduce overfitting and improve stability under varying network conditions.

7.2.2 Conclusion of Research Question 2

The results demonstrates that some parameters had a higher impact when predicting SE in order to determine the optimal threshold than others. In particular, the mean UE SRS SINR, the mean uplink pathloss, and the aggregated throughput were identified as the most influential input features in the XGBoost model. The analysis also showed that the averaged features were generally more important than their corresponding variance based features. This indicates that long-term average network conditions had a stronger influence on the predicted SE than short-term irregularities in the network measurements. In addition, the results demonstrated that no single parameter alone was sufficient to influence SE on their own, in order to determine the optimal threshold in all scenarios. Instead, the model relied on a combination of features related to radio conditions, throughput, and UE behavior, showing that the optimal threshold depends on multiple network conditions rather than a single parameter.

7.2.3 Conclusion of Research Question 3

The results show that a dynamic threshold can improve downlink performance, in terms of SE, compared to fixed thresholds in certain scenarios. Improvements in SE were observed in 48.3% of the evaluated cases at cell level, while lower SE was observed in 34.9% of the cases. Although the observed SE differences were relatively small, the results indicate that a dynamically threshold selection has the potential to improve overall network performance by adapting to varying network conditions rather than relying on static configurations. A positive net value of around 0.15 bps/Hz indicates that the dynamic model successfully outperforms the static baseline. The results also showed that the dynamic approach is sensitive to limitations in the training data. The disproportion and overlap between thresholds likely affected the model predictions and contributed to situations where a lower SE was obtained.

7.3 Future Work

Several areas for future work were identified throughout this thesis. Firstly, the data collection process. Additional datasets collected over a longer time period and during more diverse network conditions would likely improve the stability and generalization capability of the ML model. Future data collection could also benefit from the thresholds being separated into dedicated logs, where each log contains data corresponding only to a specific threshold. This would reduce the overlap between thresholds observed in this thesis and result in cleaner training data. Furthermore, evaluating a larger set of thresholds than the five considered in this thesis could provide a more detailed understanding of the relationship between threshold selection and SE. Another idea for future work is the investigation of additional ML models beyond Random Forest and XGBoost.

Further analysis of the relationship between SRS SINR and SE is also recommended. As discussed in Chapter 3, multiple crossing points were observed in the SRS SINR versus SE figures, see Figure 4.4. Although this was outside the scope of the this thesis, it represents an interesting result that may provide additional insight into the behavior of the network and the threshold selection process. Another potential improvement could be additional filtering and categorization of the dataset in order to increase the reciprocity proportion of the collected data. One example is to include information related to UE mobility. A stationary UE may benefit from maintaining a stronger connection for a longer period of time, whereas a UE moving fast through the network, such as in a vehicle, may not benefit equally from certain threshold configurations. Including mobility-related filtering could therefore improve the relevance of the training data and the resulting model predictions.

Finally, including more features could further improve the ML model. Parameters such as Timing Advance (TA) may provide additional information regarding the UE mobility and network context. TA indicates how far a UE is from the base station. Larger TA values indicate that the device is located farther from the base station [34]. Such features could help the model better capture dynamic network behavior and improve the prediction of the optimal threshold under varying conditions.

Bibliography

[1] Ericsson. Make the most of Massive MIMO. [Online]. Available: https://www.ericsson.com/en/ran/massive-mimo

[2] ——, "Massive MIMO Handbook - Extended fourth edition," *Ericsson*, vol. Fourth edition, 2025. [Online]. Available: https://foryou.ericsson.com/massive-mimo-extended-fourth-edition-registration.html

[3] J. Skäremo, "Learning from Real-World Networks: A Machine Learning Framework for Smart SRS Resource Allocation in 5G Base Stations," Dissertation, Malmö University, 2025. [Online]. Available: https://urn.kb.se/resolve?urn=urn:nbn:se:mau:diva-78176

[4] A. Iseni, "Machine Learning-based Precoder Type Selection in Massive MIMO Networks," Dissertation, Blekinge Institute of Technology, 2025. [Online]. Available: https://urn.kb.se/resolve?urn=urn:nbn:se:bth-27745

[5] P. Chaki, J. Shikida, and K. Muraoka, "Resource Allocation for Sounding Reference Signal in 5G Massive MIMO Under Channel Aging," in *2024 IEEE Globecom Workshops (GC Wkshps)*, 2024.

[6] C. Fiandrino, G. Attanasio, M. Fiore, and J. Widmer, "Traffic-Driven Sounding Reference Signal Resource Allocation in (Beyond) 5G Networks," in *2021 18th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, 2021.

[7] E. Björnson. (2025) How Cell Towers Actually Work (Base Station Tour). YouTube video, uploaded 2025. Uploaded by Wireless Future. [Online]. Available: https://www.youtube.com/watch?v=5DtwjQnNgiw

[8] M. Kihl and J. Andersson, *Datakommunikation och nätverk*, 2nd ed. Studentlitteratur, 2020.

[9] Ericsson, "Step in to a world of 5G - Fast, available, secure, reliable and capable," 2026, accessed: 2026-02-16. [Online]. Available: https://www.ericsson.com/en/5g

[10] T. Ghirmai, "Precoder selection and rank adaptation in MIMO-OFDM," in *2012 11th International Conference on Information Science, Signal Processing and their Applications (ISSPA)*, 2012, pp. 555–560.

[11] K. Bhukya, S. A. Sheikh, and R. K. Ganti, "Precoder Implementation and Optimization in 5G NR Massive MIMO Radio," in *2025 National Conference on Communications (NCC)*, 2025.

[12] K. Säfsten and M. Gustavsson, *Forskningsmetodik 2.0: För ingenjörer och andra problemlösare*. Studentlitteratur, 2019.

[13] A. Brijith, "Data Preprocessing for Machine Learning," *International Center for AI and Cyber Security Research and Innovations (CCRI)*, 10 2023.

[14] A. R M and D. Prakash, "A Simple Approach for Selecting the Best Machine Learning Algorithm," *International Journal of Scientific and Engineering Research*, vol. 12, pp. 902–909, 2021.

[15] Avik Dutta. Random Forest Regression in Python. https://www.geeksforgeeks.org/machine-learning/random-forest-regression-in-python/. Accessed: 2026-03-18.

[16] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001. [Online]. Available: https://doi.org/10.1023/A:1010933404324

[17] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD*, 2016.

[18] scikit-learn. scikit-learn Machine Learning in Python. https://scikit-learn.org/stable/index. Accessed: 2026-03-18.

[19] D. M. L. C. (DMLC). (2026) Xgboost documentation. https://xgboost.readthedocs.io. Accessed: 2026-05-06.

[20] scikit-learn. RandomForestRegressor. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor. Accessed: 2026-03-18.

[21] ——. RandomizedSearchCV. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV. Accessed: 2026-03-27.

[22] ——. GridSearchCV. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV. Accessed: 2026-03-27.

[23] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning: with Applications in Python*. Springer, 2023.

[24] S. Dhingra, "Evaluation Metrics II," *Towards Data Science*, 2021, accessed: 2026-04-13. [Online]. Available: https://towardsdatascience.com/evaluation-metrics-ii-e6f09ded4981/

[25] M. W. Ahmed. (2023, Aug.) Understanding Mean Absolute Error (MAE) in Regression: A Practical Guide. https://medium.com/@m.waqar.ahmed/understanding-mean-absolute-error-mae-in-regression-a-practical-guide-26e80ebb97df. Accessed: 2026-04-13.

[26] scikit-learn. 3.4. Metrics and scoring: quantifying the quality of predictions. https://scikit-learn.org/stable/modules/model_evaluation.html#prediction-error-display. Accessed: 2026-04-21.

[27] R. He and Y. Xu, "Overfitting identification in machine learning models with the person-fit indicator," in *2023 4th International Conference on Computer Engineering and Intelligent Control (ICCEIC)*, 2023, pp. 520–524.

[28] S. Geman, E. Bienenstock, and R. Doursat, "Neural Networks and the Bias/Variance Dilemma," *Neural Computation*, vol. 4, no. 1, Jan. 1992.

[29] GeeksforGeeks. (2025) How can Feature Selection reduce overfitting? Last updated: July 23, 2025. Accessed: 2026-04-22. [Online]. Available: https://www.geeksforgeeks.org/machine-learning/how-can-feature-selection-reduce-overfitting/

[30] SHAP documentation contributors. Beeswarm plot. Accessed: 2026-04-22. [Online]. Available: https://shap.readthedocs.io/en/latest/example_notebooks/api_examples/plots/beeswarm

[31] GeeksforGeeks. (2025) Diagnostic Plots for Model Evaluation. Accessed: 2026-05-07. [Online]. Available: https://www.geeksforgeeks.org/r-machine-learning/diagnostic-plots-for-model-evaluation/

[32] Pennsylvania State University. (2023) 4.6 - Normal Probability Plot of Residuals. Accessed: 2026-04-22. [Online]. Available: https://online.stat.psu.edu/stat501/lesson/4/4.6

[33] K. Vännman and A. Jonsson, *Matematisk statistik*, 3rd ed. Studentlitteratur, 2020.

[34] S. Mukherjee, A. K. Sinha, and S. K. Mohammed, "Timing Advance Estimation and Beamforming of Random Access Response in Crowded TDD Massive MIMO Systems," *IEEE Transactions on Communications*, vol. 67, no. 6, pp. 4004–4019, 2019.