Delft University of Technology
Master's Thesis in Embedded Systems

Indoor Localization of Bluetooth Low Energy Nodes Surrounded by Random Obstacles and Interference

Javier James

embedded software

TU Delft Delft University of Technology

This page was deliberately left blank

Indoor Localization of Bluetooth Low Energy Nodes Surrounded by Random Obstacles and Interference

Master's Thesis in Embedded Systems

Embedded Software Section
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
Mekelweg 4, 2628 CD Delft, The Netherlands

Javier James
J.A.James@student.tudelft.nl

2nd April 2020

Author
Javier James (J.A.James@student.tudelft.nl)
Title
Indoor Localization of Bluetooth Low Energy Nodes Surrounded by Random Obstacles and Interference
Msc presentation
7 April 2020

Graduation Committee

Prof. dr. K.G. Langendoen (chair)	Delft University of Technology
Dr. ir. A. van Genderen	Delft University of Technology
S. Ouborg	Kien

Abstract

The company Kien [1] is developing Bluetooth Low Energy (BLE) speakers, consisting of a subwoofer and several satellite speakers. These subwoofer and satellite speakers will have a feature named Follow Me, which when enabled from a smartphone application, will play music only in the room where the user currently is. The BLE's signals will encounter obstruction, reflection, and diffraction from obstacles such as people, walls, and furniture. These indoor propagation issues cause the received signal strength (RSS) to have more distortion and variate more. Indoor position systems (IPS) that make use of a geometric model, such as using the RSS in trilateration methods, will suffer greatly from the propagation issues. On the contrary, fingerprinting methods that use a statistical model are more robust. However, fingerprinting requires a time consuming setup that needs to be redone periodically and every time after the environment changes.

We implemented a Naive Bayes Gaussian classifier that can distinguish whether the signals travelling between a pair of BLE speakers, which are placed in two fully furnished adjacent rooms, came from speakers in the *same room* or in *different rooms*. Subsequently, the speakers that are in the *same room* can be grouped together, which is a subtask for the Follow Me feature. The classifier uses two features derived from the RSS information. Namely, the mean of the sum of several RSS samples (MSUM) and the standard deviation of the RSS range of several sample sets (SRR).

We have quantified the classifier's robustness by evaluating its performance in 9 different environments that had obstacles or radio frequency (RF) interference or both. We chose the acceptable performance to be an area under the curve (AUC) of at least 80.00 %. The classifier was able to successfully distinguish between BLE speakers that were in the *same room* and in *different rooms* in 7 environments with a minimum and maximum AUC of 86.64 % and 100.00 %, respectively.

[1] https://www.kien.io/

This page was deliberately left blank

Preface

During the course of my academic studies and working at Kien I have been exposed more to the concept of indoor localization, which has become something I enjoy. Furthermore, from when I was young robotics has always been something I found intriguing. Working at Kien has given me the opportunity to apply machine learning concepts, which are also used in robotics, to indoor localization.

The journey has been long and challenging and therefore, I would like to give thanks to everyone who has helped me directly and indirectly. First of all I would like to give thanks to the higher power that I believe in. In those moments where I felt that the work load was too much, I could centralize my energy in prayers to find hope, peace, and strength.

Furthermore, I would like to give thanks to my professor Koen Langendoen and my colleagues at Kien, especially the managers, Florent Gatin, Jorn Ouborg, and Sten Ouborg. Koen has been very patient with me for taking a longer time on my thesis than it was desired. You have also given me the opportunity to work on my thesis part-time and gave me good guidance in moments when I needed it the most.

Sten, Jorn, and Florent have helped me greatly. You have given me moral support when I felt that the combined load of work and thesis was too much to bear. Additionally, you have also given me extra days off from work to focus on my thesis, for which I am very grateful. Furthermore, you have given me the opportunity to work on my thesis part-time while working on the main development of the Kien's product on the remaining days. I am also thankful for your patience and guidance during this journey.

My colleagues have helped me with motivation, brainstorming, and assistance where needed. Vincent, thanks for the example code to begin my thesis project. Xabi, thank you also for the example code and for the brainstorm sessions. Costas, thank you also for the python scripts to make graphs for my report.

I would also like to give thanks to my family and friends, especially Ashford, Maria, Raul, Dijeannio, and Ludmilla, who have supported me. Every now and then you would contact me to see how I am progressing and doing emotionally, and you would give me encouraging words to keep pushing.

And to everyone else that I did not mention explicitly and has helped me in some way, thank you.

Javier James

Delft, The Netherlands
2nd April 2020

Contents

This page was deliberately left blank

Chapter 1

Introduction

A multi-room audio system is a centralized audio system for the home, which consists of a main speaker (hub) and several speakers. It provides the means for a user to listen and control the audio in all rooms with speakers simultaneously or individually using a mobile device such as a smartphone. Kien is a start-up company in the Netherlands that makes multi-room audio systems consisting of a subwoofer (SUB) and portable satellite (SAT) speakers.

Follow Me is a technology patented by Kien that enables a Bluetooth Low Energy (BLE) multi-room system to play audio automatically in rooms where the user carrying a smartphone is located. This thesis sets the foundation for this technology by researching a method to automatically group speakers based on the room they are in.

1.1 Motivation

Kien's speakers are expected to work out of the box, requiring minimal user interaction to set up. This entails that once the speakers are taken out of the box, they can be powered on and placed in the desired locations and the system automatically deduces what speakers are in the audio network and in the same room. The set of all speakers that are in the same room will be referred to as a *bundle*. SATs and a user's smartphone connect to the SUB in order to join Kien's multi-room audio system. Consequently, the connected SATs will be considered to be in *group mode*, as it highlighted in Appendix A. All speakers in *group mode* will play music in sync. The Follow Me feature monitors the user's location using BLE and detects within a few seconds when a user enters a room that contains speakers from the audio network. Follow Me requires mainly two separate, yet dependent stages. First, the system locates BLE speakers and deduces what speakers are in the same room. This results in a logical list of *bundles*, whereby each *bundle*

consists of a set of speakers that are in the same room. The actual room is, therefore, not important. Second, the system locates the user carrying a smartphone and determines if the user is in a room containing speakers. Subsequently, music will play only through the speakers that are in the same room with the user. This thesis focuses on the first stage.

A *node* is a connection point within a distributed network. All speakers and a smartphone behave as nodes in a BLE network. There are several ways localization is done in order to obtain the position of an unlocalized device (UD), also known as the target. Bluetooth indoor localization is usually done using methods that use the received signal strength (RSS) with a path loss model (PLM) [26] [29]. The PLM is used to capture the relation between the radio frequency's (RF) RSS and the distance between the target and a reference node. RSS methods are trivial and low cost, especially when used with already existing infrastructure [40] [48]. The simplicity of this method is preferable for Follow Me, however this method has several complications. First, the location of the reference nodes are usually known [47], whereas speakers location are not known during the setup of Kien's system. Second, the path loss model must represent the current environment, making it challenging to select a single model that works for every room [29]. Free space is an environment with no obstacles between the sender and receiver, resulting in a clear line of sight (LOS). In an environment with obstacles the RF path will be obstructed resulting in no clear LOS. The RF signal will propagate differently in different environments. Therefore, the PLM can only be used to model a specific environment [32]. Third, the RSS fluctuates more due to the signal reflecting, diffracting, and scattering on obstacles in a room [27]. This is known as the multipath effect, which describes the phenomenon where a signal will traverse from sender to the receiver via multiple paths, each experiencing different distortion, resulting in combined signal to be distorted even more by receiver. The multiple are created due to reflection, diffraction, and scattering. Fourth, the RSS may be unreliable since it may give the wrong distance when attenuated by obstacles, such as furniture and people. Obstacles cause the signal path to not have a direct path between a sender and receiver, resulting in the absence of a clear LOS [21].

Alternatively, in proximity-based methods reference nodes use the RSS to detect when the target is in close proximity by comparing the RSS to different configured thresholds [16] [21]. This method is simpler to implement than RSS methods using PLM. However, proximity-based methods will also suffer from similar multipath issues as RSS methods using PLM [40]. Therefore, this method is also challenging to provide accurate localization for Follow-me.

Furthermore, fingerprint-based methods are also used to model the RSS pattern in several areas [18] [28] [34]. Fingerprint-based methods create an RSS database of each room and statistically model the RSS patterns.

Machine learning algorithms are usually used to determine in what room a target is currently located by comparing a new RSS sample, which was sampled by the target, to the RSS database. Fingerprint-based methods are more robust to multi-path issues than RSS methods using PLM and proximity-based methods [40]. However, fingerprint-based methods have several drawbacks. First, they require a time consuming and labor intensive setup in order to create the database, which is not desirable for Follow Me. Second, fingerprint methods are not robust to indoor environment changes [19] [48]. And since Follow Me requires minimal interaction and a room's environment may change, the challenge in the application of fingerprinting is reducing the setup time and finding an optimal way to update the database whenever the environment changes.

Given the previously mentioned issues in the common BLE localization methods, it is necessary to research a more reliable method that can localize BLE nodes indoors accurately despite the obstruction caused by obstacles. Additionally, it will be crucial to find a method that requires no user interaction.

1.2 Problem Statement

Assume the environment depicted in Figure 1.1 with two adjacent rooms, the office and the living room, which are separated by a wall with a sliding door. Both rooms are fully furnished and contain nodes, *SUB1*, *SAT2*, *SAT3*, and *SAT4*, which are all connected to Kien's audio network. In the living room SAT2 and SAT3 are close to one another. SUB1 and SAT1 are in the office, whereby both nodes are placed on top of a bookshelf. The nodes in the office are placed very close to each other and to the wall that separates the two adjacent rooms. Furthermore, assume that the location of a smartphone that a user is carrying is unknown to the system and is therefore, an unlocalized device (UD).

The UD is currently in the living room, however much closer to the office's nodes. Therefore, the system might falsely localize the UD to be in the office with *SUB1* and *SAT1*. As it was mentioned in Section 1.1, RSS methods using PLM will rely on the estimated distance between nodes to determine the UD's position. Alternatively, proximity-based localization will measure the RSS and compare it to several thresholds. The wall that separates the living room from the office will attenuate the RF signals propagating between the UD and the office's nodes. In an ideal scenario, this attenuation will be strong enough to make a clear distinction between signals propagating in the office and signals propagating in the living room, despite the fact that the nodes in the office are closer to the UD. In practice however, the construction material used in walls varies in different buildings and rooms [32]. Therefore, a scenario will occur where the RF signals will not be

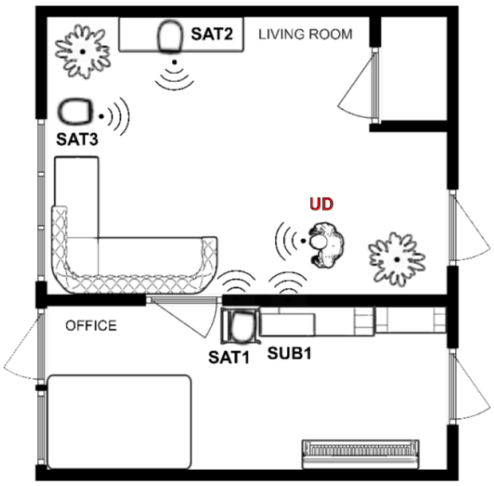


Figure 1.1: Environment with 2 adjacent rooms containing each 2 BLE nodes (speakers). The location of a smartphone that a user is carrying is unknown. Therefore, the smartphone is referred to as an unlocalized device (UD). The UD is placed in the living room, however closer to the office's nodes. The wall that separates both rooms does not attenuate all the radio frequency (RF) waves propagating from the nodes in the adjacent room. The localization system should position the UD as being in the living room, despite being close to the office's speakers. However, the system might falsely localize the UD in the office.

attenuated enough, causing the Follow Me feature to not accurately locate nodes or a user's smartphone. This is caused by the fact that the system of nodes will perceive the UD, when located close to the wall, to be in close proximity to the nodes in the office. The UD will then be falsely localized as being in the office, instead of in the living room.

In addition to the wall degrading the performance of localization methods that use RSS with path loss model or proximity, Section 1.1 also indicates that since nodes are placed randomly within the same room they are prone to be blocked by furniture or other types of obstacles. Furthermore, people that are walking in the room will obstruct the RF signals causing unpredictable attenuation and reflection. The aforementioned mentioned issues might eventually cause the localization system to localize the UD to be in the wrong room.

Much research has been done into using RF for indoor localization with RSS model-based methods. But the problem with RSS model-based methods is that they give a single model which will not be portable to other environments. Alternatively, the fingerprint method is usually used since it is more robust to the absence of LOS than RSS methods using a PLM or proximity. However, fingerprinting requires a time consuming and labor

intensive task of creating a radio-map containing the RSS data from different transmitters collected in multiple rooms. Furthermore, the radio-map needs to be updated when a major environment change occurs (e.g., moving a couch or one of the nodes) in order to maintain a good accuracy [46]. All the previous positioning methods rely mainly on anchor nodes, which are nodes whose location are known, in order to localize the UD [47].

We limit the project's scope to only finding a solution to automatically group BLE nodes together with minimal user interaction. The work carried out will be researching and designing a solution that can distinguish between BLE nodes, which are placed in two fully furnished adjacent rooms, in the ***same room*** from nodes in ***different rooms***. Furthermore, although it would be good to create a solution that can locate BLE nodes while a user move them in real-time, the focus will be dealing with a static setup. Given the complications using current localization methods for Follow Me, the research question for this thesis is:

How to accurately group Bluetooth Low Energy (BLE) nodes by the room that they are located in, while being robust to radio frequency (RF) interference, attenuation, and multipath effect caused by people and obstacles, such as walls and furniture, with zero user interaction?

The main challenges in solving the problem statement are listed below:

1. **Interference and the multipath effect**: RF devices, such as BLE nodes, encounter challenges indoors such as the multipath effect resulting in an increase in the distortion present in RSS signal [40] [48]. Obstacles, such as people, furniture, and walls cause the RF signal to attenuate, diffract, and reflect. Copies of the signal will travel through multiple path, each experiencing different phase shifts and distortion. The combined signal at the receiver will be distorted even more, causing the RSS to variate more [35].

   Additionally, signal distortion may occur more when other devices in the same frequency as BLE are nearby, such as a microwave and other BLE nodes [48].

2. **Difficult to model the RF propagation for all environments**: Different environments have different furniture layouts and different construction materials in walls [32]. Therefore, signal will propagate differently and the multipath effect will be different. It is challenging to find a model to represent the propagation for different environments [31]. Localization such as RSS-model-based and proximity-based will suffer from the multipath effect [32].

3. **Environment changes over time**: The setup of a room might eventually change. Furniture may be added, removed, or moved. This will

have an impact such that the RF will propagate differently in that room [35]. Methods that model the environment such as fingerprinting will not be robust to these environment changes.

1.3 Contribution

The main contribution of this thesis is that we have implemented a Naive Bayes (NB) Gaussian classifier that with just 2 features performs with an AUC $\geq$80 % and will function with the same performance in several other environments.

1.4 Organization

The organization of this report is as follows: First, Chapter 2 provides the necessary background information. Second, Chapter 3 highlights the state of the art research on BLE indoor localization. Third, Chapter 4 describes the classifier that was designed and the features used in order to group speakers together. Fourth, Chapter 5 highlights how the classifier was implemented and the communication infrastructure used. Fifth, Chapter 6 evaluates the classifier and discusses the results. Finally, Chapter 7 concludes the research and provides recommendations for future work.

Chapter 2

Background

This chapter provides the necessary information required to understand the topics and methods discussed in this thesis.

2.1 Machine Learning and Classifier Model

Pattern recognition is the scientific discipline that focuses on categorizing (classifying) objects, based on some measurements, to several categories (classes). A multi-class classifier categorizes objects to more than two class, while a binary classifier categorizes objects only to two classes. Every object will be defined by several observed characteristics, also known as features. Moreover, an object is referred to as *a pattern* or *an observation* and is mathematically defined as a vector $\mathbf{x}$ consisting of l features [44], as it shown in Equation 2.1.

$$\mathbf{x} = \{x_1, x_2, ...x_l\}^T \tag{2.1}$$

The feature space is l-dimensional space containing all the observations. In machine learning the goal of a classifier is to learn how the patterns for each class are categorized by means of analyzing each observation's feature (metric). The classifier creates decision lines (decision thresholds) that separates the patterns in regions, which represents the different class. Classifiers can be grouped into 3 types based on how the learning process occurs, namely, supervised, unsupervised, and semi-supervised learning. Supervised learning consists of two sequential phases, the training and then the testing phase. A class label is a label assigned to an observation in order to declare to what class it belongs to. In the training phase an observation table is created using N observations with known class labels. This creates a Nx*(l+1)* matrix that contains N observations, whereby each observation contains l features and the class label appended. The observation table will be used as training data, whereby a classifier model will trained how the behavior amongst each patters are for the different classes. Once the training phase

is completed the classifier can then classify new observations to a given class in the testing phase. This is done by comparing the features of a new observation to that of the training data and determining to what class the new observation is most probable to belong to.

Unsupervised learning differs from supervised learning in that the classifier model does not need an observation table that consists of class labels. The classifier will find similarities between the patterns and cluster similar patterns.

In semi-supervised learning a combination of supervised and unsupervised learning is applied. The classifier is trained with a set of observations for which class labels are known. Additionally, the classifier is given set of observations with unknown class labels in order to find similarities.

Pattern recognition problems normally follow a baseline sequential flow. Feature generation, feature selection, classifier design, and then system evaluation. At any given stage the process can return to any of the previous stages for improving the model performance further.

The feature generation stage involves creating features from raw measurements. This can also be done by combining features and generating a new feature, such as the product of two features. The more features that are used by the classifier, the higher the chance of experiencing *the curse of dimensionality*. The curse of dimensionality m means that the number of observations needed by the classifier to achieve an acceptable performance increases exponentially with the dimensionality of l [44]. Consequently, the computation time will be increased and the required storage space will be more.

Once the features have been generated, the feature selecting phase can begin. The feature selecting phase involves selecting the best combination of features, which gives a higher classifier performance.

Consequently, the classifier is designed and implemented. Given the input observations, consisting of features, the classifier then classifies each observation. Finally, the classifier's performance is evaluated based on some metric, as it is described in Section 2.2.

2.2 Classifier's Evaluation Methods

In the previous section, Section 2.1, it was highlighted that a classifier's performance is evaluated using some metric. The selected metric is influenced by the desired classifier optimization for a specific application and the available dataset. The main metrics used are confusion matrix, accuracy, area under curve (AUC), precision, sensitivity (recall), and specificity. The accuracy, AUC, precision, sensitivity, and specificity are all based on the confusion matrix. Additionally, the log loss is a next metric used [15].

Section 2.1 also highlights that an observation is a classifier's input sample

Table 2.1: General concept of a confusion matrix. Given N observations that were predicted the confusion matrix summarizes the total number of true positive (TP), false positive (FP), true negative (TN), and false negative (FN).

		Actual		
		Positive (1)	Negative (0)	Total
Prediction	Positive (1)	TP	FP	$TP + FP$
	Negative (0)	FN	TN	$FN + TN$
	Total	$TP + FN$	$FP + TN$	N

that contains features describing it. Furthermore, that a classifier classifies new observations to a given class. In other words the classifier predicts to what class it assumes the observation belongs to. For this thesis the observations define samples from the Bluetooth Low Energy (BLE) radio frequency (RF) signals between a pair of nodes. We focused on two classes, namely *same room* and *different rooms*, therefore we made use of a binary classifier. The class *same room* and *different rooms* are represented by the classifier's discrete outputs 1 (positive class) and 0 (negative class), respectively. In a binary classifier application the positive class defines the desirable class, which is the class we are focused on getting correct.

Confusion Matrix The confusion matrix is a foundational performance metric. Table 2.1 highlights that the confusion matrix consists of 4 terms, true positive (TP), false positive (FP), true negative (TN), and false negative (FN). For this thesis, the positive class with the classifier's output value 1 represents an observation in the class *same room*. Whereby, the class *different rooms* is represented by the negative class with the classifier's output value 0. Given a set of N observations that a classifier have classified, the confusion matrix summarizes the total number of TP, FP, TN, and FN.

When an observation that actually belongs to nodes from the same room is accurately predicted to belong to the class *same room* then the classifier's response is defined as a true positive (TP). However, if the observation was inaccurately predicted as belonging to nodes in *different rooms* then the prediction was a false negative (FN). A FN is also known as a *miss* or ***type 2 error***.

On the other hand, when an observation belongs to nodes from different rooms, if the classifier accurately predicted the observation to be in *different rooms* then the classifier's response is labeled as a true negative (TN). Otherwise, the prediction is inaccurate and defined as a false positive (FP). A FP is also known as a *false alarm* or ***type 1 error***.

Sensitivity The sensitivity, also known as the recall or the true positive rate (TPR), is a metric used to evaluate how well a classifier can detect an observation that belongs to a pair of nodes that are actually within

the same room and accurately predict the nodes to be in the class *same room*. The sensitivity is derived by evaluating how many times the classifier accurately predicted observations to be in the *same room*, from the set of all observations that belonged to nodes from the same room. Equation 2.2 illustrates how the sensitivity is calculated.

$$recall = TPR = \frac{TP}{TP + FN} \tag{2.2}$$

When nodes are actually in the *same room*, higher sensitivity implies that a classifier will accurately predict observations in the class *same room* more. Consequently, the *type 2 error* will be decreased.

To the contrary, lower sensitivity implies that when nodes are actually from the *same room*, the classifier will miss it and inaccurately predict the observation to be in *different rooms*, resulting in the *type 2 error* increasing.

It is desirable for this thesis to not miss detecting when 2 nodes are actually in the same room. Therefore, it is preferred to optimize the classifier for high sensitivity and thus minimizing the *type 2 errors*.

The rate at which a classifier misses the detection of observations from the class *same room*, which results in an increase in the *type 2 errors*, is defined as the false negative rate (FNR). Equation 2.3 describes how the FNR is mathematically defined and shows that the FNR is negatively related to the TPR. The more frequent a classifier inaccurately predicted observations as *different rooms* while they were actually from nodes in the same room then the higher the *type 2 errors* and FNR will become. The inverse holds true.

$$\begin{aligned} FNR &= \frac{FN}{TP + FN} \\ &= 1 - recall \end{aligned} \tag{2.3}$$

Specificity The specificity, also known as the true negative rate (TNR), is a metric that describes how well the classifier don't respond to observations that don't belong to the desirable class, which is *same room*. In other words, given all observations that belong to nodes that were in different rooms, the specificity highlights how often the classifier accurately predicted the observations to be in *different rooms*. Equation 2.4 highlights how the specificity is mathematically defined.

$$specificity = TNR = \frac{TN}{TN + FP} \tag{2.4}$$

The higher the specificity is then the classifier will inaccurately predict observations to be in the *same room* less often, when they actually belong

to nodes from *different rooms*. Consequently, the *type 1 errors* (FP) will decreased.

On the other hand, the lower the specificity is then the classifier will inaccurately predict observations to belong to *same room* more often, when they actually belong to nodes from *different rooms*. The decrease in specificity results in the false positive rate (FPR) increasing, as Equation 2.5 highlights. The FPR is negatively related to the specificity and positively related to the *type 1 error* (FP).

$$\begin{aligned} FPR &= \frac{FP}{TN + FP} \\ &= 1 - specificity \end{aligned} \tag{2.5}$$

Sensitivity and Specificity trade-off If the data distribution of all the classes are completely separable then a classifier can easily achieve high sensitivity and specificity. However, if the data distribution of the classes overlap errors will be made. Therefore, a trade-off between sensitivity and specificity must be taken. Given a (slightly) overlapping distributions, if a classifier adjusts its decision line such that more of the observations that belong to nodes from the *same room* are detected then the classifier's sensitivity (TPR) will be increased and the *type 2 errors* decreased. Consequently, the FNR will be decreased. However, more observations that belong to nodes from *different rooms* will be inaccurately classified as being in the *same room*. Therefore, the *type 1 error* (FP) will increase, resulting in an increase in the FPR and a decrease in the specificity.

To the contrary, if the classifier's decision line is set such that less observations from nodes in the same room are detected then the classifier's specificity (TNR) will be increased. This is a result of an increase in true negatives (TN). Therefore, there will be less *type 1 errors*, which results in a less false positive rate (FPR). However, the classifier will inaccurately predict observations to be in the class *different rooms* more often while they actually belong to nodes in different rooms. Therefore, there will be an increase in *type 2 errors* (FN) and thus the false negative rate (FNR). Since the true positive rate (TPR) is negative relates to the FNR, the TPR will be decreased. That is, the classifier will detect less frequent when observations belong to nodes from the same room [13].

Area Under Curve (AUC) The receiver operating characteristic (ROC) is a 2D graph that plots the TPR versus the FPR for every possible classifier's decision line possible. A ROC curve is created, as it is shown in Figure 2.1. The area under curve (AUC) is the area under the ROC curve and is a performance metric that is often used. The AUC summarizes the overall ratio between TPR and FPR amongst every given threshold. Therefore,

the AUC declares the classifier's overall capabilities to distinguish between classes, independent of where the final decision line was set.

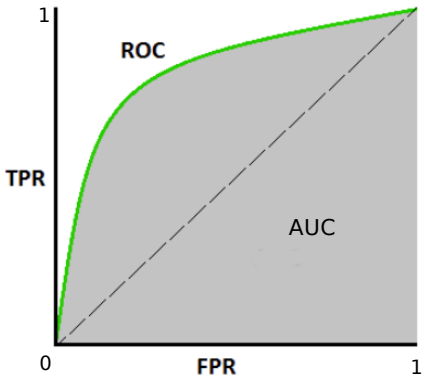


Figure 2.1: The area under the curve (AUC) for a region of convergence (ROC). The ROC-AUC graph shows a the ratio of the true positive rate (TPR) versus the false positive rate (FPR) for different decision thresholds. The diagonal line defines an AUC of 0.5 where a model just randomly predicts. The closer the AUC is to 1, above the diagonal line, the better the model can distinguish between positive and negative class. In the case of an AUC below the line, the model makes the opposite prediction from what the actual class is. Image edited from [13]

For our binary classifier the AUC will declare how well the classifier distinguishes observations that belong to BLE nodes in the *same room* from those in *different rooms*. The AUC values range between 0 and 1 and is dependent on the data distribution of the observations for each class in the feature space. An AUC of 1 indicates that a classifier can perfectly separate observations from nodes in the *same room* from those in *different rooms*. However, in practice an AUC of 1 is difficult to achieve. Moreover, an AUC of 0.5 describes a classifier that is as good as randomly predicting an observation to any class. This is shown in Figure 2.1 as the diagonal line and it defines a non-informative model. An AUC can be a also be a result of overlapping data distributions. An AUC of 0 indicates that a classifier completely inverses its output predictions. The classifier will inaccurately predict all observations to be from nodes in the *same room*, while they belong to nodes from *different rooms*. The inverse holds true [13]. For this thesis the AUC will be expressed in percentage and thus ranging between 0 % and 100 %.

Log Loss The log loss is a metric used to determine the level of uncertainty and checks how confident a classifier was when it inaccurately predicted an observation. In other words, a classifier that made a wrong prediction is penalized more if it was more confident about its prediction.

Moreover, the log loss requires that the classifier provide the predicted

probability that an observation belonged to a specific class. The predicted probability is then compared to the classifier's discrete output of the actual classes. The formula for a binary classifier is given in Equation 2.6. Given a binary classifier with 2 discrete outputs, namely *same room* and *different rooms*, the classifier's output is defined as $y \in$0,1. For each observation the classifier can predict with a probability $p = Pr(y = 1)$. For an observation that belongs to the class *same room*, namely the positive class with $y = 1$ the log loss becomes $L_{log}(1, p) = -ylog(p)$. The closer the prediction is to 0, declaring that the observation belongs to the class *different rooms*, the more the prediction will be penalized. The closer the prediction is to 1 the less it will penalized. The contrary holds true for observations belonging to the class *different rooms*. The formula will be $L_{log}(0, p) = -log(1 - p))$, The prediction will be penalized more the closer the probability is to 1, declaring that the observation belongs to the class *same room*. The closer the observation is to 0, the less it will be penalized.

The log loss values range between 0 and ∞. The closer the log loss is to 0 the better. The log loss is influenced by two factors, the number of classes and if the data is balanced. For a binary class that is balanced the prevalence of any of the class is maximum $p = 0.5$, which will mean that an acceptable log loss is $L(1, 0.5) = -log(0, 5) = 0.69$ [15]. For an imbalanced binary classifier, with the positive class having a prevalence of p_1 , the acceptable log loss is $L(1, p_1) = -log(p_1)$

$$\begin{aligned} L_{log}(y, p) &= -logPr(y|p) \\ &= -ylog(p) + (1 - y)log(1 - p) \end{aligned} \tag{2.6}$$

This page was deliberately left blank

Chapter 3

Related Work

The state of the art indoor localization methods will be analyzed and critically evaluated in this section. The purpose is to learn how other researcher deals with the localization challenges similar to the one stated in Section 1.2.

3.1 System's Requirements and Research's Scope

In order to develop a system that provides a good practical solution for Kien's Follow-me feature, it is important to specify what requirements are needed for the Kien's system. Consequently, the requirements will set the context and scope of this literature review and thesis research. The system's requirements are as follows:

- REQ-1: Room-level accuracy to localize and group/bundle an unlocalized device (UD), which will be a BLE speaker, according to the room it is in;
- REQ-2: To automatically detect that speakers are in the same room;
- REQ-3: Minimal user interaction/set up cost;
- REQ-4: Responsive, such that when an UD is placed in a new room that the system is able to determine to which speaker *bundle* the UD belongs to;
- REQ-5: Energy efficient;
- REQ-6: Scalability, such that by adding or removing speakers from a room or to the network, the system will still be able to detect a UD in a room with speakers;
- REQ-7: Robust to interference and the multipath issues caused by obstacles, mainly from walls, furniture, and people;

- REQ-8: Flexible positioning of the BLE speakers.

Given these requirements this thesis and literature review will mainly focus on indoor localization Bluetooth Low Energy nodes. The BLE nodes are mainly considered to represent BLE speakers. Special attention will be given to how different solutions satisfy the requirements in terms of performance, scalability, robustness, node positioning, and user interaction. Additionally, only papers dating from 2009 to recent will be reviewed.

3.2 Wireless Positioning Techniques

In most application localization will be done by means of wireless communication. Generally this require for the target to transmit a wireless signal which a receiver can then pick up to deduce the targets current position. The target may transmit Wi-Fi, Bluetooth (Classic or Low Energy), Acoustic, RFID, GPS, etc. Indoor localization is the research that is focused on methods to locate a target indoors and mainly uses some of the previously mentioned wireless communication, excluding GPS. Indoor localization can mainly estimates a person's location by sampling the signals between the mobile phone that a user is carrying and anchor nodes. In addition to the previously mentioned signals, sometimes a hybrid method is used to enhanced the overall performance, overcoming any limitation from any individual signal [47].

Wireless positioning is the localization of an unlocalized device (UD) with the use of a wireless system. A sensor node is a node in the network that can gather sensory information, process data, and communicate with other nodes in that network. The location of sensor nodes may be unknown. Anchor nodes, also known as base stations, are (sensor) nodes that locations are known [17], which can be determined using GPS [47]. The techniques used to accomplish wireless positioning are mainly done by estimating the distance between the UD and the anchor nodes. Wireless tracking differs from wireless positioning in that wireless position deduces the location of the UD once, while wireless tracking does this continuously.

Localization done outdoor achieve high accuracy when using GPS or cellular systems. However, these two systems have a degraded accuracy indoors due to the absence of a line of sight (LOS). Indoor localization require therefore other methods that uses other wireless signals.

Indoor localization uses mainly the following wireless signals: Wi-Fi, Bluetooth (Classic or Low Energy), Acoustic, and RFID. Apart from the wireless signals there are embedded sensors in devices, such as a mobile phone, that are mainly used for indoors. Namely inertial sensors and camera.

There are several measurements that can be attained from the wireless signal between the infrastructure with base stations (BS) and the target.

These measurements are used to deduce the target's current position. Figure 3.1 shows the taxonomy of techniques used for wireless localization. Localization can be grouped based on the measurement it uses and what technique is used to deduce the target's position. These methods differ also on the necessary infrastructure that is required. Localization methods can be further grouped how the observed data is modeled and used with respect to the anchor nodes position to deduce the target's position.

Some measurements are calculated using time and relates to the range or the difference of ranges, respectively, time of arrival (TOA) and time difference of arrival (TDOA). TOA is a example of the Trilateration technique and TDOA of the type Multilateration. Moreover, there are measurements related to angle such as the angle of arrival (AOA), which is of the type Triangulation. Consequentially, there are measurements related to the the signal strength such as the received signal strength (RSS). The RSS measurement can be used in 3 different ways. First, the distance between the target and several nodes are calculated using a path loss model (PLM). Then, trilateration is used to deduce the target's absolute position. RSS models using a PLM and trilateration will be referred to as RSS-model-based localization. Second, statistical measurements of the RSS is used in the statistical models such as Fingerprint methods or Bayesian methods. Fingerprint methods will be referred to as fingerprint-based localization. Fingerprint-based localization deduces the target's symbolic location. Third, the RSS can be used to detect close proximity of the target, and will be referred to as proximity-based localization. Proximity-based localization estimate the target symbolic location to an anchor node. A path loss model is also used to estimate the distance [48]. Finally, there are measurements which are related to the quality of the signal in a digital communication channel such as the link quality indicator (LQI) and the bit error rate (BER).

The following section will focus on describing how localization methods can be grouped based on an observation model $\mathbf{h(x)}$, which describes how the position of base stations and the target relate to an observed measurement $\mathbf{r}$. The generic form of the observation model is described in Equation 3.1. Given an infrastructure with M base stations, $\mathbf{r}$ defines the observed measurement from each base station, $\mathbf{r} = \{r_1, r_2, ...r_M\}$. The vector $\mathbf{x}$ is the target's coordinates. The function $\mathbf{h(x)}$ defines the observation model that maps the target position, and indirectly incorporates the base stations' position, to the observed measurements. Finally, $\mathbf{e}$ defines the measurement's error. Localization methods differ in how the observation model is defined and how the target position is deduced [40]. Localization methods can be grouped in three main groups, namely, geometric model, proximity model, and statistical model.

$$\mathbf{r} = \mathbf{h(x)} + \mathbf{e} \tag{3.1}$$

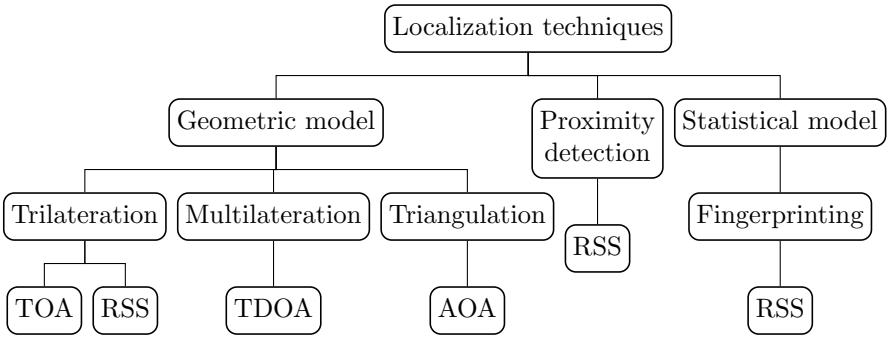


Figure 3.1: Taxonomy of wireless localization techniques [47].

3.2.1 Geometric Models

Geometric models estimate the position of the target by means of applying some sort of function such minimizing a cost function as described on Equation 3.2. Depending on the measurement used the observation model is defined differently [40].

$$V(\mathbf{x}) = \log p_e(\mathbf{r} - \mathbf{h(x)}) \tag{3.2}$$

Trilateration The trilateration method uses the absolute distance between the UD and several anchor nodes to deduce the UD's position. See Figure 3.2. The distance between the UD and a given anchor node can be visualized as the radius of a circle with the anchor node at the center. This entails that the UD can be anywhere on that circle, as it is shown in the image that illustrates the usage of one anchor node, A. When multiple anchor nodes are used the intersection of the circles illustrates the UD's possible position(s). In a scenario with two anchor nodes, A *and* B , the target node's location reduces to two possibilities. It requires at least three anchor nodes in order to reduce the UD position to one possibility. Therefore, trilateration requires at least three anchor nodes [47].

Trilateration makes use of time of arrival (TOA). Additionally, the RSS measurement is used in RSS-model-based localization. TOA and RSS-model-based differ from each other on how they estimate the distance between the anchor node and the UD [47].

TOA methods use time to estimate the absolute distance. RF signals propagate through the air with the speed of light. The distance can be deduced from the time it took for a signal to traverse between the target node and the anchor node. Equation 3.3 highlights how the distance for each of the M anchor nodes are calculated to create the observation model [40].

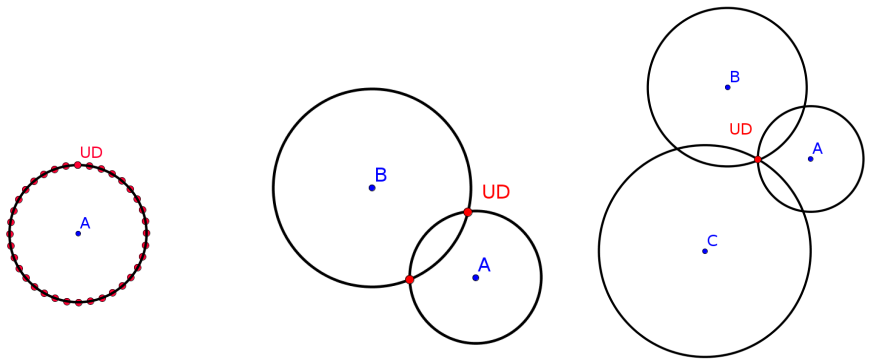
$$\mathbf{h(x)} \ = ||\mathbf{x} - x_i||, i = 1, ..., M \tag{3.3}$$

The process begins with the target node sending a signal to each anchor node with the time of flight (TOF). Each anchor node then records the time of arrival (TOA). This method requires the UD and all anchor node to be time synchronized. This timing requirement is a challenging task and requires special hardware that may cost a lot.

A RSS path loss model gives the relation between the RSS and distance. RSS-model-based localization relies on a path loss model (PLM) to estimate the distance between the UD and an anchor node. Equation 3.4 highlights how the the distance estimation is done for each of the M anchor nodes, whereby RSS_0 highlights the RSS transmitted by the $i-th$ anchor node. The variable α_i highlights the path loss exponent between the target and the $i-th$ anchor node.

$$\mathbf{h(x)} = RSS_0 - 10\alpha_i \log ||\mathbf{x} - x_i||, i = 1, ..., M \tag{3.4}$$

In comparison to TOA methods, no synchronization is needed, therefore costs are reduced. However, there are several challenges with RSS-model-based localization. First, it is difficult to find a model to accurately characterize the real environment [47]. Second, a model is unique only to the environment it was tuned for [47]. Third, just like TOA methods, RSS-model-based methods suffer from the multipath effect [40] [46].



(a) One anchor node *(b) Two anchor nodes* *(c) Three anchor nodes*

Figure 3.2: Localization using the trilateration method. In Figure 3.2a one anchor node with known location is used, namely node A, The unlocalized device (UD) may be located on any point on the circle surrounding A. This circle will have a radius equivalent to the distance between the UD and the anchor node. Figure 3.2b highlights that when using two anchor nodes, namely the nodes A and B, the UD may be in either of the two intersection. However, Figure 3.2c shows that when three anchor nodes are used, namely the nodes A, B, and C, the UD will be located by the single point, which intersects the radius from all anchors [12].

Multilateration Multilateration is similar to trilateration in the sense that both technologies use the distance between the UD and several anchor nodes [47]. However, the essential concept of multilateration is that it additionally uses the difference in distance between each pair of nodes to deduce the UD's location. This is illustrated in Figure 3.3. Multilateration differs from trilateration in that it uses relative distances instead of absolute distances. Using the distance between the UD and a given anchor node and the inter-node distance a hyperboloid can be visualized for each pair of nodes. The UD can be anywhere on that plane. When multiple pairs of nodes are used, several hyperboloid are present which may intersect. Similar to trilateration, the intersection defines the possible location of the UD and at least three anchor nodes are necessary to have one possible location.

A technique which uses multilateration is Time Difference of Arrival (TDOA). This technique uses time to derive the required distances just like TOA, and its observation model is also defined the same way as shown in Equation 3.3 [40]. TDOA requires timing synchronization on all anchor nodes, however the UD does not need to be in synchronization. This makes TDOA cheaper than TOA. However, faces the same challenges with the multipath effect indoors.

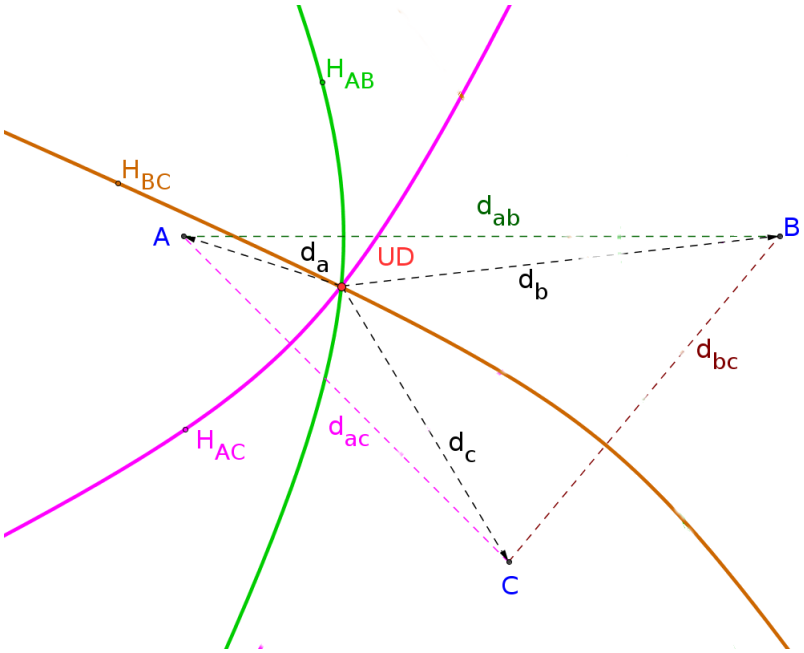


Figure 3.3: Localization using the multilateration method. The unlocalized device interacts with the three anchor nodes, A, B, and C, to deduce its position. First, the UD determines the distance between each anchor node. Subsequently, each anchor node communicates with each other and calculates the difference in distance/time. This creates a hyperbola, H_{ij}, between node i and j [47].

Triangulation Triangulation methods differ from lateration by using angles instead of distances. An example is Angle of Arrival (AOA). This method requires the angle to be calculated at which a signal from the UD reaches anchor nodes [47], respectively node A and B as it is shown in Figure 3.4. An inter-node measurement is also needed to determine the distance between the anchor nodes, and the observation model is defined as that of Equation 3.3 [40]. At least two nodes are required to deduce the UD's position. The AOA method suffers from large position estimation errors when small measurement distortion occur [47].

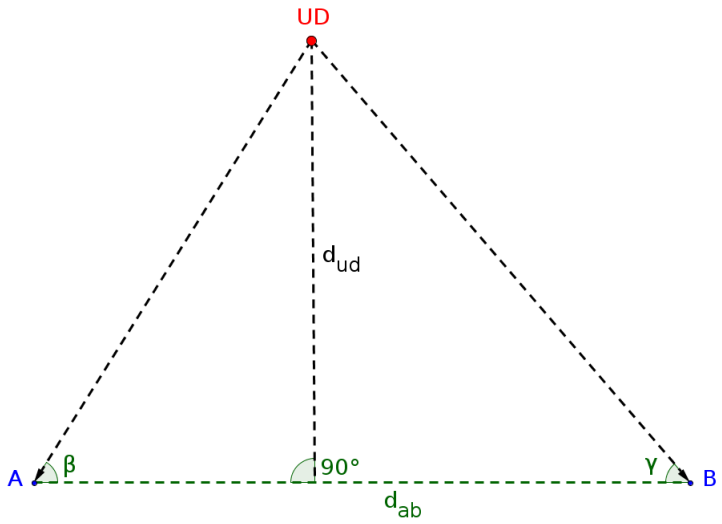


Figure 3.4: Triangulation method [12].

All of the previously mentioned localization methods are computation efficient. However, they do not work well in situation of NLOS since it can not detect outliers caused by NLOS well [46] and the localization accuracy is degraded. The localization accuracy can be improved by taking multiple samples and then reapplying the localization method over each sample, which will result in multiple estimated positions. Subsequently, an estimator robust to noise, such as the mode, is used over all the estimated positions in order to get the target's position. This method to improve the localization accuracy has a trade-off that the localization accuracy will improve at the expense of a higher computation cost and energy consumption.

3.2.2 Proximity

Proximity-based methods rely solely on detecting the presence of the UD near a given anchor node. This implies that the UD needs to be in close proximity. Proximity-based methods also use a path loss model to estimate the distance, however, mainly to create a *geofence* [48]. The geofence

represent symbolic locations based on threshold indicators. The use of symbolic locations implies that the target actual position is not estimated. The simplest form of indication will be labeling the proximity to be far or near.

Reference nodes are sorted by their distance as seen from the UD perspective. Consequently, the location of the anchor node with the closest distance is selected to target's location.

Proximity-based localization is very simple, but also suffer from indoors multipath effect. To improve the localization accuracy indoors the density of the reference nodes need to be increased, which in return will increase the infrastructure cost.

3.2.3 Statistical Models

Another type of localization technique is fingerprint-based methods. This method differs from Trilateration and Triangulation methods in that fingerprint-based methods do not calculate the distance between the target and anchor nodes. This method analyzes the environment and deduces the target's position by finding the best match of a new sampled fingerprint to a radio-map created during calibration [19].

During this scene analysis the raw RSS values are sampled as the input metric. A fingerprint is a vector $\mathbf{r}$ containing RSS measurements from different anchor nodes as they are received by the UD for a given location, such as a room in a building. A radio-map is a database that consists of fingerprints from all the areas in which the localization is desired. Given an environment with M anchor nodes and N locations the observation model is described in Equation 3.5, whereby $p_i(\mathbf{x}_j)$ defines the data distribution of the measured signals from the $i-th$ anchor node at the location x_j.

$$\mathbf{h(x)} = p_i(\mathbf{x}_j), i = 1, ...M, j = 1, ..., N \tag{3.5}$$

Fingerprint-base localization consists of two phases, the off-line and the on-line phase. In the off-line phase, also known as the scene analysis phase, the radio-map is created by recording the observed measurements from all M anchor nodes in all of the N locations.

In the on-line stage a UD will be located by sampling a new fingerprint measurement and comparing it to radio-map to see which area has the best match. This is done mathematically by means of Equation 3.6. The estimated position, $\hat{\mathbf{x}}$, is the $j-th$ location that minimizes the error defined by z_j. Furthermore, r_i defines the new measurement received from the $M-th$ anchor node. The parameter $p_i(\mathbf{x}_j)$ defines the model created in the off-line stage for the $j-th$ location, as it described in Equation 3.5.

$$\hat{\mathbf{x}} = \text{argmin } {z_j}^2$$
$${z_j}^2 = \sum_{i=1}^{M} (r_i - p(x_j))^2 \tag{3.6}$$

In order to find the best match machine learning algorithms are used, such as k Nearest Neighbor (kNN) and Bayesian [40]. Other forms of machine learning that have been used in fingerprint-based methods are neural networks, support vector machines, and decision trees [40]. When using kNN the k estimated position that provides the lowest error z_j are used and averaged to deduced the final estimation of the target. Bayesian algorithms are used to deduce the target position by means of determining the maximum likelihood estimator of the position, as it is highlighted in Equation 3.7.

$$\hat{\mathbf{x}} = \text{argmax } p(r_1, ..., r_n | x_j) \tag{3.7}$$

Equation 3.8 describes how the target's position is calculated using the Bayes rule.

$$p(x_j | r_1, ..., r_M) = \frac{p(r_1, ..., r_M | x_j) p(x_j)}{p(r_1, ..., r_M)} \tag{3.8}$$

The observation model is defined by $p(r_1, ..., r_M | x_j)$, which is the probability that the measurements from the M anchor nodes were observed in the $j - th$ location. Additionally, $p(x_j)$ describes the prior probability for each location and $p(r_1, ..., r_M)$ the probability of observing the received sample from each of the M anchor nodes.

Fingerprint-based methods can be cheaper than the trilateration, multilateration, and triangulation methods since existing infrastructure can be reused and no special hardware is needed for synchronization [19]. However, creating a radio-map is time consuming and an exhausting task that needs to be done in the off-line stage before localization can happen. Furthermore, this method's accuracy degrades over time as changes to that environment causes the radio-map to model the actual environment less accurately. The off-line phase needs to be repeated again to compensate for this [46]. Moreover, a downfall of fingerprint-based methods is that the target's position can only be estimated in the areas where data measurements have been recorded in the off-line phase [40].

Despite the challenges in fingerprint-based localizations, it mostly used for methods that use RSS since it is robust to outliers caused by NLOS scenarios indoors [40] [20] [47].

This page was deliberately left blank

Chapter 4

System Model

RF signals propagating through a room between a sender and a receiver may experience reflection, diffraction, and attenuation from the furniture, walls, and other obstacles in the room. Since each environment is different, it is expected that the signals will propagate differently [31] [35]. In [35] the challenge was that for different environments the standard deviation varied significantly. Nevertheless, we chose to analyze the standard deviation in different room, since it may be distinguishable amongst different environments. Therefore, we focused on taking into account the statistics of the signal's variation when generating most of the features.

Fingerprint-based methods has shown to be robust to several of the indoor challenges mentioned in Section 1.2 [40] [47]. Therefore, we also implemented a method that analyzes the environment and applies statistical algorithms, which fingerprint-based method do. On the contrary to using the raw RSS metric in fingerprint-based methods, our method makes use of other metrics generated from the RSS information. Moreover, most of the generated metrics incorporate analysis of the statistical properties of the signal fluctuation.

It was highlighted in [47], that errors always occur in fingerprint-based systems, using WLAN, although reasonable accuracy. This is because in fingerprint-based methods different locations have similar radio signature, which is caused by the dynamic propagation of radio signals. Since BLE and WLAN are in the 2.4 GHz, this was taken into account by aiming to find features that model the signal's random behavior even more. The goal was to model the signal fluctuation in a room and determine a way to uniquely distinguish between two BLE nodes exchanging BLE packets from within the ***same room*** from those in ***different rooms***. Recall that in Section 1.1 it was highlighted that Follow Me requires to know what speakers are in the *same room*, thus in the same *bundle*. Since the classifier will only classify whether a pair of nodes are in the *same room* or in *different rooms*, the classifier results need to be further processed in order to create the *list of bundles*. First, the number of distinguishable groups must be identified.

Second, every node that are in the same room must be categorized to the same bundle. The creation of the *bundle list* is not done for this thesis and it was for future work.

The remainder of this chapter focuses on the metrics (features) that were used to model the BLE signals propagation indoors. Additionally, this chapter describes the classifier that was designed and implemented to classify pairs of BLE nodes to being in the *same room* or in *different rooms*.

4.1 Feature Generation and Selection

The purpose of this phase was to find the features that describes the two classes, *same room* and *different rooms*, in such a way that there is a clear distinction between the BLE signals traversing between a pair of BLE nodes that are in the *same room* and *different rooms*.

A single RSS measurement is referred to as a sample, whereby several samples is referred to as an RSS chunk. The RSS signal between a pair of BLE nodes is captured over time to create a group of *RSS chunks*, which are then processed to create a single observation. Some features process the group of RSS chunks using a non-overlapping sliding window, processing each RSS chunk individually. The results of each processed RSS chunk are then further processed to create the final single value feature. Other features, on the other hand, process the entire group of RSS chunks to create the single value feature.

The chosen features can be classified in the way a set of BLE signal is analyzed, resulting in two types. Method 1 focuses on modeling the RSS by analyzing its fluctuation. An RF signal propagating in a clear Line of Sight (LOS) in a room is not expected to experience much attenuation [48]. However since a room in a home consists of objects, the impact of the multipath effects on the RSS fluctuation will be different [31] [48]. A portion of that RF signal may penetrate the surrounding walls and can therefore be observed by neighboring nodes in adjacent rooms [32] [40] [46]. This signal will be attenuated differently due to different wall materials, and is expected to fluctuate more than a propagation path with a LOS. The RF fluctuation in a static environment comes to a certain equilibrium and hardly changes after a while [35][27]. The fluctuation rate has shown to be unique per environment[35]. RF signal fluctuation has therefore been chosen to be one of the main feature. Features of this type are the *Mean of the Set of RSS chunks' Standard deviation using RSS (MSSR)*, *Fluctuation Rate Distribution (FRD)*, and *Standard deviation of the set of RSS chunks' Range (SRR)*.

On the other hand method 2 focuses on a variety of approaches that do not focus on the signal fluctuation. Features belonging to this group are *Mean of the set of RSS chunks' Interquartile Range using RSS (MSIQRR)*,

Percentage Of Peaks (POP), and the *Mean of the set of RSS chunks' Sum (MSUM).*

4.1.1 Mean of the Set of RSS Chunks' Standard Deviation using RSS (MSSR)

Motive The standard deviation of the RSS focuses on analyzing how much the RF signal strength variated from the average signal strength over time. Recall that in Section 1 it was stated that the multipath causes the RSS to experience more distortion and resulting in more RSS variation. [27] refers to this as fading. Analyzing the standard deviation of a RSS chunk, consisting of several RSS measurements,, can highlight if the signal is experiencing fade depth.

A low standard deviation means that the signal strength does not variate much over time, giving some sort of indication that the RF path may not be under obstruction. This can be the result of the absence of obstacles or that the RF path between a receiving and transmitting nodes have reached an equilibrium [27]. Additionally, that the nodes are at a static position. The higher the std_rss is the more multipath fading the RF signals may have experienced. This can be due to moving objects within the room, obstacles between the nodes, such as a wall or furniture, or that the nodes have been placed to a new location.

The standard deviation is applied over several RSS chunks over time. To smooth out the overall variation amongst the RSS chunks we apply the mean over all RSS chunks. The mean is robust to outliers, however, if the fade depth changes continuously over time, the MSSR is expected to increase. For as far as we know, we have not noticed such a feature used for fingerprint-based methods.

Limitations The fluctuation is expected to be relatively a lot more in a room that is occupied versus a room that is not. This increase in fluctuation can be similar to the fluctuation that the RF link encounters through a wall, which make this feature challenging in occupied areas.

Since the variation of the signal strength is used, it is not expected that this feature will be affected much by the distance between sender and receiver. To the contrary, it is expected to be affected more by the obstacles that are between the receiver and the sender. However, the mean and standard deviation are not robust to outliers. This feature also works best when the shape of the data is normally distributed having a bell curved shape

4.1.2 Fluctuation Rate Distribution (FRD)

Motive The Fluctuation Rate Distribution (FRD) focuses on the rate at which the RSS of a new packet have variated from the previous received

packet. The distribution of this fluctuation rate over time is then analyzed.

This algorithm was researched in [35] and shown to be able to detect the presence of a person in a room. We have implemented the FRD algorithm in software. [35] highlights that each room has a unique rate at which the RF signals propagate within them. This part is helpful for this thesis in order to differentiate between rooms. Additionally, [35] highlights that in the presence of a person the RF signals fluctuates more with smaller fade depths. This algorithm checks how much a received packet RSS value variated from the packet received at the previous timestamp. This difference in variation is captured over time to create the fluctuation rate distribution. An environment without a person always has 90% of its distribution within the ranges -1 and 1. In the presence of people the fluctuation is more and therefore also the spread of the distribution. The percentage of the distribution within the ranges -1 and 1 remains below 65%. The feature FRD defines the percentage of the distribution that is within the ranges -1 and 1.

It is expected that when there are changes in the environment that the RSS will fluctuate more compared to the previous timestamp. Moreover, it is expected that if the obstruction of a wall causes the RSS to fade even more, which also results in more fluctuation. In both scenarios the spread of the fluctuation rate distribution will be more, and therefore the FRD value decreasing then this will cause the fluctuation rate to increase.

Limitations The FRD has the advantage to be able to detect the presence a person in a room. Additionally, it is a feature that has been claimed to uniquely distinguish different rooms. However, if a room is occupied, it is expected that the FRD may have similar characteristic to an RF path obstructed by a wall or other obstacles.

4.1.3 Standard deviation of the set of RSS chunks' Range (SRR)

Motive This standard deviation of the RSS range (SRR) takes a sample set and calculates the range, $R = \max(RSS) - \min(RSS)$. This feature focuses on the spread of the RSS in a RF path. A RF path that is obstructed tend to have a larger spread. Fading is also caused by multipath effect and by shadowing of obstacles, such as the presence of people.

Limitations The SRR provides a simple way to know if there is an obstruction in the RF path. However, this feature is quite similar to that of the Interquartile Range (IQR), however it is not robust to outliers. So the standard deviation was taken over time to try to minimize the variation in the result.

4.1.4 Mean of the Set of RSS Chunks' Interquartile Range using RSS (MSIQRR)

Motive The interquartile range (IQR) of the RSS is a measure to see how spread out the RSS distribution is and what percentage of the signal's distribution is within 1 standard deviation from the median. Using the IQR and the median is similar to using the mean and the std, but unlike the mean and the std, the IQR and median are robust to outliers.

The IQR of the RSS indirectly states that approximately 50 % of the signal's strength were within the quartiles Q1 and Q3, respectively -1 std and +1 std from the median. A sample set, which contains RSS samples between two nodes, with a higher IQR is expected to indicate that the RF path was less stable. The larger the IQR the more fade depth the RF signal has encountered. It is expected that the IQR will be impacted more by the RSS variation caused by reflection than the distance between nodes.

The stability of the RSS can indirectly be shown by means of a box plot A box plot shows the skewness of a sample set's distribution. If the distribution is skewed then it means that certain RSS values have been received more than other values, which indirectly indicates less variation. Additionally, when the IQR is small then it indicates that the data samples are not often widely spread apart from one another. Therefore, highlighting that smaller fade depth have occurred. The rate and amount of variation is expected to be difficult to see in the IQR directly on the RSS distribution.

After observing the IQR of several samples and RF paths, it has been observed that the IQR seems to have some sort of relation to the distinction between nodes within the *same room* versus those from the *different rooms*. For as far as we know, we have not noticed such a feature used for fingerprint-based methods.

Limitations IQR is a measure of central tendency and the spread of the data, while being robust to outliers. Additionally, it is robust to the skewness of the distribution. IQR can be used to find outliers and know the skewness of the distribution in an easier manner than using the mean and standard deviation.

4.1.5 Percentage of Peaks (POP)

Motive It has been observed that at times when nodes are in the *same room* the observed RSS between each RF path tend to have some similarities in the shape of the RSS distribution. BLE nodes that were in the *same room* seemed to have an RSS distribution that was more narrow and had less and higher peaks. RF paths that had more variation, for example when nodes were in separate rooms was expected to have a RSS distribution that was more broad and also with smaller and broader peaks. For as far as we know,

we have not noticed such a feature used for fingerprint-based methods.

Limitations This feature is not robust to outliers.

4.1.6 Mean of the Set of RSS Chunks' Sum (MSUM)

Motive This feature takes the RSS chunk a sample and calculates the sum of the RSS value. This feature focuses on the sum of the signal strength, to capture the RSS during a period time even if it has been variated for a period of time. A RF path that is obstructed tend to have a weaker RSS. Over time, if obstructed long the sum of the RSS will be really low. However, if the RF path was blocked for a short while, the sum will be still lower than normal. Furthermore, if nodes are far apart it is expected that the sum of the RSS will be really low. The closer the nodes are, the higher the *MSUM* will be.

Pros and Limitation The MSUM provides a simple way to know when speakers are far or close to one another. However, this feature is not robust to outliers. Furthermore, if nodes from another room is close by, there will be no easy way to distinguish that the node is behind a wall.

4.1.7 Features Evaluation

The accuracy of a model is strongly influenced by the selected features. Having too many irrelevant features will increase the complexity, training time, and may cause over-fitting. Over-fitting is when a classifier to take account noise instead of relevant patterns and will therefore decrease its performance [6].

The desire is to have features that are correlated with the output variable, *same room* and *different rooms*. Features that are not correlated with output are considered irrelevant and can cause the model to over-fit. That is, the model will be trained for details and will not be generalized. Therefore, during the testing phase new samples can be wrongly classified.

On the other hand, if two relevant features are used, which are highly correlated, then one will be redundant. This will increase the complexity of the classifier without much performance gain [44]. An increase in software complexity will cause more computation , resulting in more power consumption.

Feature selection is the art of selecting the best combination features that result in good classifier's performance. Irrelevant or redundant features may cause a model to decrease in performance. Classifiers require generalization. The higher the ratio between the number of patterns and features, the less the classification error will become [44]. This section highlights how the final features were selected.

The goal was to find features that have high discriminatory information with respect to the classifier's output, while reducing the number of features. Additionally, determining what combination of features provide the best performance. The feature space is the scatter plot of the observation with the features as the dimensions. In the feature space features should have large distance values between classes and small variance between the values within the same class [44].

A Random Forest classifier was implemented in Sci-kit learn, using the default settings, in order to select the best combination of features to use for the final classifier. The training data with all of its features for the environments *people walking* and *minimal room* were used. A Random Forest classifier is built upon several Decision Tree classifiers. A Decision Tree classifier is a nonlinear classifier, which consists of nodes that represents the features, capable of grouping features based on how well they reduce the node impurity. The Gini impurity is a type of node impurity that defines the probability of misclassifying a new observation if the observation was randomly classified according the current data distribution, containing the class labels. The lower the Gini impurity is the better a feature can split the dataset such that it contains more samples of a class compared to the other class. In other words, the features are split based on which features separates the dataset best with respect to the classes *same room* and *different rooms*. The selected features using a Decision Tree classifier, however, will may over-fit. The Random Forest classifier improves the over-fitting by generalizing the selecting features. This is done by using an ensemble of Decision Trees and using as the input to each tree a randomly selected subset of the features and dataset. The Random Forest assigns a score which defines a feature relative contribution to the decision making. The higher the feature importance score is, the more the feature contributes to the decision making in the Random Forest [44] [9].

Table 4.1 highlights the features important sorted from top to bottom with descending order of importance. The two top features are *MSUM* and *SSR*. Therefore, we chose these two features to be used in the final classifier.

4.2 Bluetooth Low Energy Speakers' Room Classifier

4.2.1 Models' Evaluation

Section 2 highlights the performance metrics used for evaluating a classifier's performance. The main metric for this thesis is the area under the curve (AUC), since this metric highlights how well a classifier distinguishes between observations from BLE nodes in the *same room* and *different rooms*.

Appendix B highlights how several classifier models were trained and eval-

Table 4.1: Feature importance using a Random Forest classifier on the test only dataset. The features are giving a score representing the average level of decreasing the Gini impurity and thus statistically having a stronger relationship with the classifier's output values same room and different rooms.

Feature	Random Forest Score
MSUM	0.66
SSR	0.13
MSIQR	0.08
MSSR	0.06
FRD	0.04
POP	0.03

uated to select the top 3 classifiers. The result of the classifiers evaluation is shown in Table 4.2. The performance metric is sorted from left to right in the order of importance. All metrics use % format except for the log loss. The log loss requires that the lower the number to indicate a better classifier's performance. The classifiers are sorted from top to bottom in the order of best performance. Overall, the top 3 classifiers were Naive Bayes Gaussian, Ada-Boost, and then Random Forest.

Table 4.2: Classifiers evaluations with respect to different performance metrics. All metrics use % format except for the log loss. The top three classifiers are highlighted in gray.

Classifier	Performance					
	AUC	F1-score	Recall	Precision	Log Loss	Accuracy
NB Gaussian	98.53	98.18	100.00	96.43	0.11	98.36
Ada-Boost	97.06	96.43	100.00	93.1	0.22	96.72
Random Forest	96.68	96.3	96.3	96.3	0.15	96.72
Decision Tree	95.59	94.74	100.00	90.00	1.70	95.08
Quadratic Discriminant Analysis	93.36	92.59	92.59	92.59	0.19	93.44
Gradient Boosting	92.97	92.31	88.89	96.00	0.45	93.44
KNeighbors	91.88	90.91	92.59	89.29	0.70	91.80
Linear Discriminant Analysis	87.42	85.71	77.78	95.45	0.2	88.52
NuSVC	80.39	76.6	66.67	90.00	0.32	81.97

The rest of this section evaluates the top three classifiers with respect to some of the system's requirements mentioned in Section 3.1, in order to select the best classifier that satisfies the over all requirements. The requirements of interest are related to how fast the classifier can classify new observations to ensure a good response time and the computation cost to insure proper energy management. Table 4.3 shows the evaluation. The

classifier are sorted from left to right and ranked by the leftmost having the highest AUC performance. With respect to the training phase, a Random Forest classifier has the highest upper-bound time complexity, whereby the NB Gaussian classifier has the best. For the prediction (testing) phase it is not simple to compare NB Gaussian with Ada-Boost and Random Forest, since NB Gaussian is not tree-based. Ada-Boost, however, predicts faster than Random Forest. Nevertheless, NB Gaussian was selected as the final classifier, since it simple to implement and has achieved a higher AUC than Ada-Boost and Random Forest.

Table 4.3: Complexity evaluation of the top three classifier models. The complexity is expressed using the Big-O notation. Each criteria is expressed against some of the system's requirements. N = number of samples, l = number of features, t = number of trees, d = tree depth and c = number of classes [14] [11] [4] [36] [3]
.

Criteria	Requirement	Classifier model		
		NB Gaussian	Ada-Boost	Random Forest
Time complexity (Training)	REQ-4, REQ-5	$\mathcal{O}(N * l)$	$\mathcal{O}(N * l^2 * t)$	$\mathcal{O}(N * l * d * \log(N))$
Time complexity (Testing)	REQ-4, REQ-5	$\mathcal{O}(l * c)$	O(t)	$\mathcal{O}(d * t)$

4.2.2 Naive Bayes (NB) Gaussian Classifier

The Naive Bayes Gaussian classifier is a very simple supervised classifier that is based on the Bayes rule [19]. The Bayes rule is shown in Equation 4.1 and defines a way to calculate the probability of a given event, *class* w_i given an observed pattern x. $P(w_i)$ defines the class probability and $p(\mathbf{x})$ the feature distribution. Finally, $p(\mathbf{x}|w_i)$ defines the class-conditional. For this thesis there are two classes *same room* and *different rooms*. Respectively class 1 and class 2 defined as w_1 and w_2. The Bayes rule provides an optimal way to reduce the classifier error in predicting an observation to be in either of the two classes wrongly [44]. The feature vector $\mathbf{x}$, $x_1, x_2, ..., x_l$, describes a pattern for an observation with values for the corresponding values for each of the l features. The variable l defines the dimension of the feature space, which is the number of features used in an observation. The class probability $P(w_i)$ and class-conditional probability density function $p(\mathbf{x}|w_i)$ are expected to be known prior to evaluating a classifier. Since this was not known, they were estimated using the training data.

$$
\begin{aligned}
P(w_i|\mathbf{x}) &= \frac{p(\mathbf{x}|w_i)P(w_i)}{p(\mathbf{x})}, i = 1, 2 \\
p(\mathbf{x}) &= \sum_{i=1}^{2} p(\mathbf{x}|w_i)P(w_i) \\
P(w_i) &\approx \frac{N_i}{N}, i = 1, 2
\end{aligned}
\tag{4.1}
$$

The central limit theorem declares that the probability density function (pdf) of the summation of random variable tends to approximate a normal distribution as the number of samples tends to infinity, independent of the pdf of the variables [44] [2]. The Gaussian Naive Bayes makes used of this theorem and can therefore assume that the class-conditional probability $p(\mathbf{x}|w_i)$ is of the type normal distribution, as it is described in Equation 4.2. The lxl covariance matrix, Σ_i, describes the covariance of each feature for that given class. It is defined as $\Sigma_i = \mathbb{E}[\ (\mathbf{x} - \mu_i)\ (\mathbf{x} - \mu_i)^T\ ]$, whereby i denotes to which of the two classes. In order to ensure that the determinant of the covariance matrix is non zero it is required that the matrix be nonsingular, thus that the features are independent. This independent assumption makes the classifier naive and has the benefit that the Bayes rule can be simplified to Equation 4.3.

$$
p(\mathbf{x}|w_i) \simeq \frac{1}{\sqrt{(2\pi)^l \det(\Sigma_i)}} \exp(-\frac{1}{2}(\mathbf{x} - \mu_i)^T \Sigma_i^{-1}(\mathbf{x} - \mu_i)), i = 1, 2 \tag{4.2}
$$

$$
\begin{aligned}
P(w_i|x_1, x_2, ..., x_l) &= \frac{P(w_i)p(x_1|w_i)p(x_2|w_i)...p(x_l|w_i)}{p(\mathbf{x})} \\
&= \frac{P(w_i)}{p(\mathbf{x})} \prod_{j=1}^{l} p(x_j|w_i), i = 1, 2
\end{aligned}
\tag{4.3}
$$

The Bayes classification rule is a simple equation that is focused on minimizing the probability error or risk [44], whereby the probability error was used for this thesis. The training data's feature space containing N samples with an observation $\mathbf{x} = \{x_1, x_2, ..., x_l\}$ was split in two regions, one for each class using Equation 4.4. The Gaussian Naive Bayes classifier can predict to which class a new observation belongs to by classifying the observation to class w_1 if $P(w_1|\mathbf{x}) > P(w_2|\mathbf{x})$. Otherwise, classify the observation to the class w_2 if $P(w_1|\mathbf{x}) < P(w_2|\mathbf{x})$. The Bayes classification rule is shown in Equation 4.5. The decision boundary is a hyperplane that orthogonal to μ_1-μ_2. The hyperplane is situated closer to μ_1 if the class probability $P(w_1) < P(w_2)$. Otherwise, if $P(w_1) > P(w_2)$ the hyperplane is closer to μ_2. However, if

$$
P(w_1|\mathbf{x}) - P(w_2|\mathbf{x}) = 0 \tag{4.4}
$$

$$\mathbf{x} \in w_i = \begin{cases} \mathbf{x} \in w_1 & \text{if } p(\mathbf{x}|w_1)P(w_1) > P(\mathbf{x}|w_2)P(w_2) \\ \mathbf{x} \in w_2 & \text{if } p(\mathbf{x}|w_1)P(w_1) < P(\mathbf{x}|w_2)P(w_2) \end{cases} \tag{4.5}$$

In Scikit-learn the Gaussian Naive Bayes classifier has only two parameters. Namely, *prior*, which is the class probability $P(w_i)$. Additionally, *var_smoothing*, which is variance added to smooth the variance across all features. The main parameters *prior* was derived from the training data and therefore, this classifier was not tuned and used the default value of *var_smoothing*.

The Naive Bayes Gaussian classifier is fast, simple, and has a small computational complexity [19]. However, if the assumptions are not met this classifier can perform poorly [44].

This page was deliberately left blank

Chapter 5

System Setup and Model Implementation

This chapter focuses on describing the communication infrastructure that was implemented and used with the BLE nodes in order to collect the necessary data samples for the NB Gaussian classifier. Furthermore, this chapter will describe the NB Gaussian classifier's parameters, which were used for the final implementation.

5.1 Communication Infrastructure

Figure 5.1 shows the high-level infrastructure of the system. Every BLE node communicates with neighboring nodes by means of advertisements, thus not an actual BLE link connection. This approach was chosen to take advantage of the BLE protocol that allows broadcasting a message to many BLE scanners at the same time. In order for the system to determine whether nodes are in the *same room* or *different rooms*, nodes advertise BLE packets containing the necessary information. The subwoofer advertises Scalable Lightweight Time Synchronization Protocol (SLTP) packets, which will be discussed more in Section 5.2. The satellites advertises Altbeacon packets. Section 5.3 will discuss the packets transmitted by satellites.

Every receiving node parses the transmitted packets from neighboring nodes and creates a local dataset. Table 5.1 highlights how each receiving node records the parsed packets into a local dataset. Note that each local dataset is recorded from the perspective of the receiving node. The *Transmitter ID* is the ID of the node that transmitted the received packet and the *RSS* is the received signal strength of the received packet. Moreover, the *TOA* is the time of arrival of the received packet. The TOA is set to be the estimated SUB's local time (elapsed time) as seen from the receiving node's perspective. The TOA will be discussed in more detail in Section 5.2. Finally, the parameter *Receiver ID* is used to identify the node that

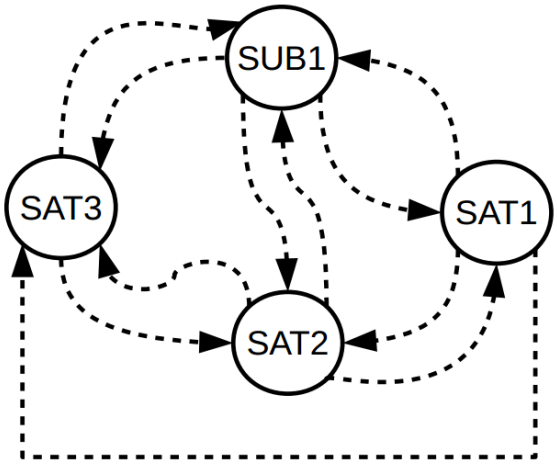


Figure 5.1: Communication infrastructure. Nodes are advertising packets, while simultaneously scanning advertised packets from neighboring BLE nodes. The SUB advertises scalable lightweight time synchronization protocol (SLTP) packets. The SATs advertise Altbeacon packets.

received the packet. The parameter *Receiver ID* comes in handy when the local dataset from each node will be concatenated in a combined dataset to form one meaningful information. The classifier can then identify the pair of nodes and process the corresponding information describing the signal traveling between the sender and receiver.

Table 5.1: Data structure for the local dataset logged in each individual node. The Transmitter ID is the ID of the node that transmitted the SLTP or Altbeacon packet. The RSS is the received signal strength of the current packet by the receiver. The TOA is time of arrival (TOA) of the received packet. The TOA is set to be the estimated SUB's local time (elapsed time) as seen from the receiving node's perspective. Finally, the receiver ID is the ID of the node that received the packet.

Parameter	Description
Transmitter ID	Node's ID that transmitted the packet
RSS	Received signals strength of the packet
TOA	Time of Arrival of the packet. Set as the SUB's estimated elapsed time
Receiver ID	Node's ID that received the packet

The local dataset that was created by each node is centralized and concatenated to form a combined dataset, which will then be used to generate the features of interest, as it is described in Section 4.1. An observation table is then created that can be used by the classifier. Section 5.5 will discuss more on the implementation of the observation table. For our experimental setup we extracted the local dataset from each node by implemented a software that allows a node to send its local dataset through a serial port.

Moreover, we connected a PC to each node's serial port and extracted their local dataset. We then made use of Python to concatenate the local datasets for further processing, which are explained in Section 5.4.

Table 5.2 highlights the main configurations for the satellite and the subwoofer. It is important to highlight that the subwoofer only transmits synchronization packets, SLTP packets, containing its local time, while scanning for Altbeacon packets from satellites. The satellites on the other hand, advertise Altbeacon packets while scanning for both Altbeacon and SLTP packets, respectively from neighboring satellites and the subwoofer.

Table 5.2: Satellite's (SAT's) and subwoofer's (SUB's) BLE configuration. A SAT advertises Altbeacon packets every 100 ms, while a SUB advertises scalable lightweight time synchronization protocol (SLTP) sync packets every 100 ms.

Description	Value
Transmission power	0 dBm
Scanning interval	600 ms
Scanning window	200 ms
Scanning timeout	never
SAT's advertisement interval	100 ms
SAT's advertisement timeout	never
SUB's synchronization interval	100 ms

5.2 Scalable Lightweight Time Synchronization Protocol (SLTP)

The BLE nodes that we implemented in the field work collectively to accomplish a specific task, which is locating a target's current position, by means of wireless communication with one another. This requires for each node to query the local time that it received packets. Subsequently, as it was stated in Section 5.1, the collected data from each node should be fused together to a single meaningful data. Therefore, it is crucial to synchronize the nodes.

Wireless sensor nodes (WSN) application and the BLE nodes used for this thesis share several things in common. First, WSN nodes may be mobile. Although we have not implemented mobility, it is taken into account in the time synchronization protocol evaluation for future improvement. Second, energy efficiency is essential. Third, nodes are distributed and must take into account being robust to obstacles and nodes not working. Therefore, we analyzed time synchronization protocols also used for WSN and focused mainly on the metrics mobility, energy efficiency, and robustness.

The requirements in Table 5.3, provides the overview of what the entire application must comply to. The time synchronization implementation

contributes to the requirements REQ-1, REQ-3 to REQ-5, and REQ-7 to REQ-9. The main metrics that are required is to have a protocol that is energy efficient and robust to obstacles. Although BLE nodes are not tracked in real time for this thesis, mobility is also taken into account for future improvement of the system. Additionally, it is good to have a protocol that has high precision.

Table 5.3: Requirements for a time synchronization protocol. The requirements are ordered using the MoSCoW analysis. M = Must have, S= Should have, C= Could have, and W= Won't have.

ID	Requirement	Priority
REQ-1	To operate in an Indoor Environment	M
REQ-2	To automatically determine what BLE nodes are in the same room	M
REQ-3	To support flexible positioning of portable BLE nodes	M
REQ-4	An energy efficiency of 10 hours	M
REQ-5	To be scalable to support between 3 to 6 BLE nodes	M
REQ-6	Zero user set up/interaction cost.	M
REQ-7	To be operational, despite if a BLE node joins or leaves the network on the fly	S
REQ-8	To be robust to interference from indoor obstacles, mainly walls, furniture, and people	C

5.2.1 Time Synchronization Methods' Evaluation

In order to select a proper time synchronization for Kien we classified time synchronized protocols based on how nodes synchronize with one another. There are mainly two categories, namely, Sender-Receiver (SR) and Receiver-Receiver (RR) [23] [30] [43].

In Sender-Receiver (SR) synchronization protocols a reference node periodically transmits a message with its local clock timestamped in it. Receiving nodes then synchronize their local clock with the reference node. This is done by using the timestamp and calculating the time delay. FTSP, LTS, TPSN, TSync, Tiny-Sync & Mini-Sync (TS/MS), SLTP, and GTSP protocols are from the SR category. SR protocols are often used and are able to receive higher precision than RR methods.

Receiver-Receiver (RR) protocols on the hand, require for the receiving nodes to synchronize with each other. Furthermore, it is not required for the reference node to advertise a reference time. Receiving nodes will record the time they received the broadcast packets from the reference node and then compare their reception times amongst other receiving nodes in order to synchronize with each other. RBS is a known protocol of this category.

Table 5.4 compares several protocols among the previously mentioned metrics. Namely, mobility, energy efficiency, robustness. Additionally, the precision is also evaluated. The metrics are organized in descending order

of importance. All two categories have protocols that are energy efficient. However, Receiver-Receiver methods do not support mobility nor are they robust. SR methods mostly have a high energy efficiency and support mobility. Therefore, SR protocols were the main focus for this thesis.

Table 5.4: Time synchronization protocol evaluation. The protocol metrics are sorted from left to right in descending order of importance [23] [38].

Category	Protocol	Performance metric			
		Mobility	Energy efficiency	Robustness	Precision(μsec)
SR	SLTP [39]	Yes	High	Low	0.13 $\pm$0.06
SR	FTSP [37]	Yes	High	Low	0.50
SR	LTS [45]	Yes	Low	Medium	$5.00 * 10^5$
SR	GTSP [42]	No	High	Medium	4.00
SR	Tiny-Sync & Mini-Sync [41]	No	High	Medium	9.45
RR	RBS [24]	No	High	Low	1.85 $\pm$1.28
SR	TPSN [25]	No	High	Low	16.90
SR	TSync [22]	No	Medium	Medium	29.00

From the Sender-Receiver category only LTS, SLTP, and FTSP support mobility, therefore the other protocols will not be used. LTS in comparison to FTSP does not have a better energy efficiency nor better precision. However, LTS is more robust. SLTP in comparison to LTS is not robust, however it does have a higher energy efficiency. Additionally, SLTP has a better precision than both LTS and FTSP, therefore we chose SLTP for this thesis. We achieve mobility, high energy efficiency, and high precision at the cost of not having robustness. This implies that a higher software layer should aim to solve this issue of robustness.

5.2.2 SLTP Implementation

SLTP is a protocol that supports clustering with multiple cluster groups. We make use of only 1 SUB and several SATs in a small environment, therefore we implemented only 1 cluster group, 1 cluster head, and several cluster members. The SUB was implemented as a cluster head and sends SLTP sync packet ever 100 ms. SATs were configured as cluster members that buffers SLTP packets over time. Subsequently, linear regressions is used on the oldest 32 SLTP packets in order to estimate the SUB's local time (elapsed time). The estimated SUB's local time is then used by SATs as their time of arrival (TOA), as it is described in Sections 5.1 and 5.3.

We re-used a structure similar to that of an Altbeacon packet to implement the SLTP packet, whereby Table C.1 highlights the main parameters. Namely, the beacon code and the beacon ID. The beacon code is a ID used to differentiate the SLTP packet from other packet. The beacon ID consists of only the *Speaker ID* and the *Elapsed time*. The *Speaker ID* is the ID of

the transmitter of the SLTP packet, which is the SUB, and is configured at runtime. The *Elapsed time* is the SUB's local time (elapsed time). Appendix C.1 highlights in more detail the modified structure of the SLTP packet.

5.3 Altbeacon packets

The Altbeacon packet was implemented mainly for satellite nodes to transmit. Altbeacon was chosen since it is an open source standard for BLE beacon packets and it may be used for both Android and iOS devices [8]. The convenience of an easy-to-implement and cross-platform protocol makes Atlbeacon useful for Follow Me, since users will have phones with different mobile operating systems.

Table C.2 highlights the main fields of the Altbeacon's data structure. Only the beacon ID and beacon code are of interest for localization. The beacon code consist of an identifier used to distinguish the packet from other BLE packets, such as the SLTP packet. The beacon ID consists of the *company universally unique identifier (UUID)*, *speaker ID*, and the *room ID*. The *company UUID* is an identifier used to identify the nodes as Kien speakers. Moreover, the *speaker ID* is used to uniquely identify each node in the system. The *speaker ID* is derived from the hardware chip's serial number.

Section 4 highlights that for Follow Me nodes are grouped in bundles to indicate which are together in the same room. Each Satellite will be advertising to what bundle it is in with respect to the SUB using the *room ID* parameter. By default all satellite advertise the value *0* for the parameter *room ID* to indicate that they are in the same room as the SUB. Once the actual bundle is determined by the system, every satellite node will get notified their updated bundle ID. Consequently, satellite noes will then advertise their new *room ID*. Recall that in Section 4 it was highlighted the classifier output must be further processed to create the *list of bundles*, which is phase not implemented in this thesis. Appendix C.2 highlights in more detail the data structure of the Altbeacon packet.

5.4 Test Runs and Creating the Observation Table

Each experiment was executed 4 times and each test run lasted between 3 to 5 minutes. As it was mentioned in Sections 5.2 and 5.3 the SUB sends SLTP sync packets and scans for Altbeacon packets from neighboring SATs. SATs on the hand, advertise Altbeacon packets, while scanning for SLTP and Altbeacon packets. Despite the packet type each node samples the RSS value from the received packet in order to create its own local dataset. The structure of a sample is shown in Table 5.1.

The features described in Section 4.1 are created by extracting the RSS measurements over a set of RSS chunks. For each pair of sender and receiver node a non-overlapping sliding window (W_{obs}), of size 80 samples is applied to the received RSS measurements.(W_{obs}) is referred to as a set of RSS chunks, since it will be processed even further in chunks.

For the feature *MSUM* the set of RSS chunks are processed using a non-overlapping window N_{window_msum} of 10 samples, creating 8 RSS chunks. For each RSS chunk the sum is calculated and then the mean is applied over all the intermediate sums to create *MSUM* value.

The feature *SSR* is also processed in a similar manner, using a non-overlapping sliding window size N_{window_ssr} of 10 samples, creating 8 RSS chunks. For each RSS chunk, the range of the chunk is calculated. The standard deviation is then calculated for each intermediate RSS ranges to create the *SSR* value.

5.5 Naive Bayes Gaussian Classifier Implementation

The Gaussian classifier from the sci-kit learn library has two parameters as it shown in Table 5.5. The class prior $P(same\_room)$ and $P(different\_rooms)$. No values were given to this input in order to ensure they were estimated from the dataset. Additionally, the default value was used for var_smooth, which is a variance added for smoothing. Equation 4.2 highlights the mathematical model for the Naive Bayes Gaussian classifier. The generic form of the mean and covariance matrix uses for this this is highlighted in Equations 5.1 and 5.2, whereby i defines the class *same room* or *different rooms*. The μ_i defines the mean and σ_i the covariance matrix of the $i-th$ class. Furthermore, σ^2 defines the correlation of a feature and σ the cross correlation of 2 features.

$$\mu_i = [\mu(SSR), \mu(MSUM)] \tag{5.1}$$

$$\Sigma_i = \begin{pmatrix} \sigma^2(SSR) & \sigma(SSR, MSUM) \\ \sigma(SSR, MSUM) & \sigma^2(MSUM) \end{pmatrix}, \tag{5.2}$$

The classifier was created by training it using the dataset from the environment *minimal room* and *person walking* combined. Recall that in Section 4.2.1 it was stated that 70 % of the data was used for training and the remainder for testing. The mean and covariance matrix for dataset from the *same room* class is depicted in Equation 5.3 and 5.4, respectively. For the class *different rooms* the mean and covariance matrix is shown in Equation 5.5 and 5.6 respectively.

$$\mu_{same\_room} = [3.68, -604.61] \tag{5.3}$$

Table 5.5: Naive Bayes Classifier's configuration

Description	Value
$P(same\_room)$	≈34 %
$P(different\_rooms)$	≈66 %
var_smoothing	1e-9

:

$$\Sigma_{same\_room} = \begin{pmatrix} 1.53 & -18.56 \\ -18.56 & 2520.98 \end{pmatrix} \tag{5.4}$$

$$\mu_{different\_rooms} = [4.47, -728.77] \tag{5.5}$$

$$\Sigma_{different\_room} = \begin{pmatrix} 2.98 & -18.35 \\ -18.35 & 1201.62 \end{pmatrix} \tag{5.6}$$

Chapter 6

Classifier's Performance

After implementing the Naive Bayes (NB) Gaussian classifier its performance was evaluated to see how well it distinguished between BLE nodes that were in the same room from those in different rooms. The AUC metric was used, since it is a metric of separability between the different classifier's outputs, which in this experiment was *same room* and *different rooms*. For a given environment the acceptable performance is 80 %. Section 1.2 indicates that the classifier should be portable such that it works in different home environments. To validate the reliability of the NB Gaussian classifier being portable it has been agreed with Kien that the acceptable performance should be met in at least 7 out of the 9 environments.

6.1 Experimental Setup

6.1.1 Environments

The remainder of this section describes the environments used to evaluate the classifier's performance. All of the experiments' setups are described in Appendix D. Figure 6.1 and 6.2 depict examples of environments that were used, respectively *adjacent room closer* and *person walking*. These were the environments where the classifier had the worst and best performance, respectively.

All environments can be grouped by the test scenario they captured. The 3 test scenarios were *placement*, *obstacles & disturbances*, and *people*. The test scenario *placement* main purpose was to understand the impact that different distances between nodes have on the performance. Moreover, the test scenario *obstacles & disturbances* took into account the impact that different type of obstacles in a home may have on the performance. Additionally, the impact of RF disturbances is also analyzed. Finally, the test case *people* focused on the impact people have on the performance.

Four speakers were used to collect samples from each environment, one SUB and three SATs. In all of the evaluated environments at least one

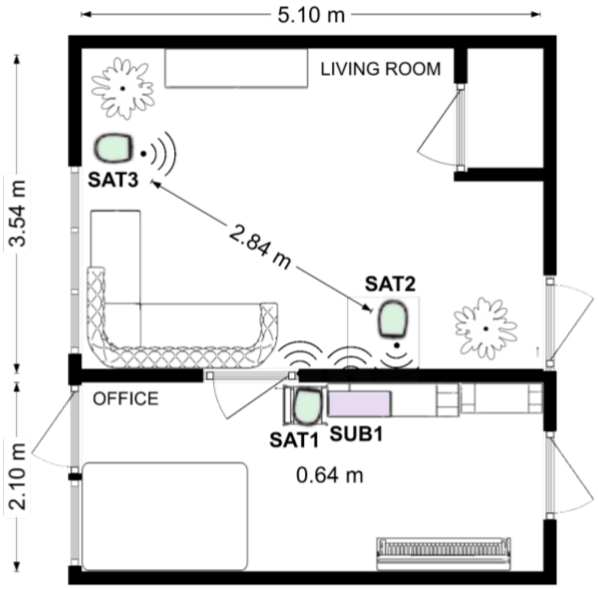


Figure 6.1: Setup of the environment adjacent room closer with 4 BLE nodes, namely SUB1, SAT1, SAT2, and SAT3. SUB1 and SAT1 were in the office and separated by a concrete wall from the nodes in the living room, SAT2 and SAT3. Moreover, SAT2 was closer to nodes from the adjacent room in comparison to SAT3, which was in the same room

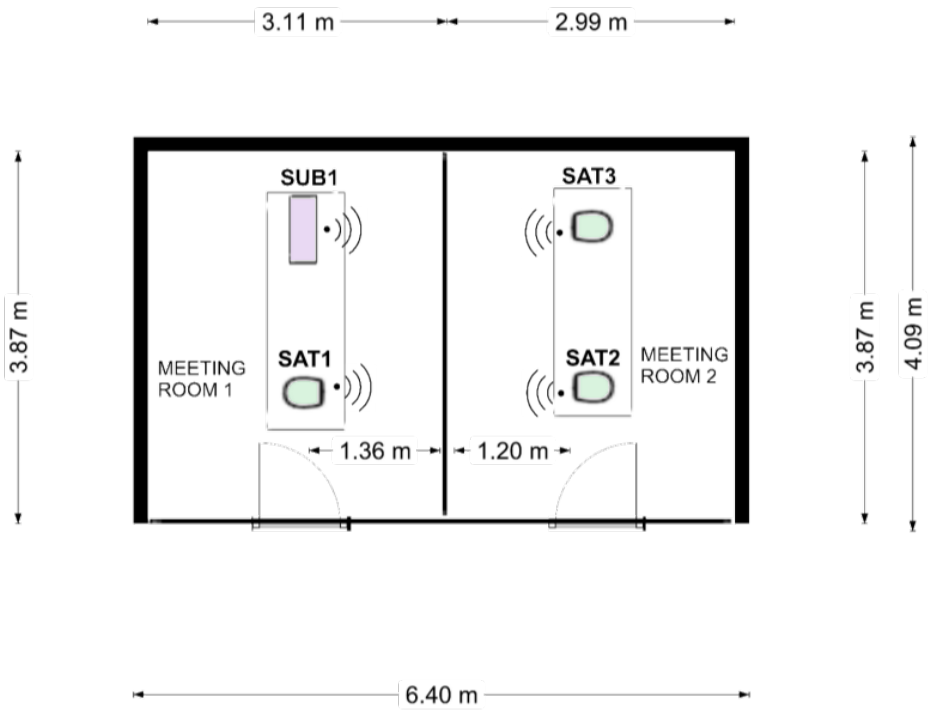


Figure 6.2: Setup of the environment person walking with BLE nodes separated by a glass wall. Two nodes were in both adjacent rooms. A person walked in, around, and then out of meeting room 2 several times.

speaker was located in one of each adjacent rooms. Table 6.1 summarizes the minimum and maximum distances between any two given nodes, whether they were in the *same room* or in *different rooms*. Section 1.2 highlights a scenario where an UD might be inaccurately classified to be together with nodes in a *different room* when the UD is closer to the nodes in a *different room* instead to the nodes in the *same room*. In other to keep track how well the classifier handles such scenarios Table 6.1 takes into account the parameters $\min d_{ij}$, $\max d_{ij}$, and Δ. Consider a pair of nodes, namely node i and j, in a given environment. The parameter $\max d_{ij}$ defines the maximum distance amongst any pair of nodes for any class, *same room* or *different room*. On the other hand, the parameter $\min d_{ij}$ records the minimum distance. In other to know if a scenario occurred, during our experiments, where a BLE node is closer to another node in a *different room* than to nodes in the *same room* we recorded the parameter Δ. The parameter Δ defines the difference between $\min d_{ij}$ for nodes in *different rooms* and $\max d_{ij}$ for nodes in the *same room*. When Δ is negative it means that there exist a node that is closer to other nodes in a *different room* than to the nodes in the *same room*. Additionally, we made use of the parameter Δ to know for what values of $\Delta$the classifier performs well or not.

Table 6.1: For all environments the distances (d_{ij}) in meters between any two nodes, namely node i and j, are highlighted. Given any two nodes the minimum and maximum distance are shown, whether they are in the same room or different rooms. The difference in distance between $\min d_{ij}$ *and* $\max d_{ij}$*, respectively from nodes in different rooms and same room, is given by* Δ*.*

		Distances [meters]				
		same room		different rooms		
Test scenario	Environment	$\min d_{ij}$	$\max d_{ij}$	$\min d_{ij}$	$\max d_{ij}$	Δ
Placement	Multiple floors	0.80	1.22	3.00	3.10	1.78
	Multiple rooms	0.64	0.64	3.7	8.80	3.6
	Adjacent room closer	0.64	2.84	0.50	3.00	-2.34
Obstacle & disturbances	Plaster wall	0.62	1.65	3.20	3.30	1.55
	Concrete wall	0.64	0.64	3.54	3.70	2.9
	RF disturbance	1.10	3.31	5.50	6.17	2.19
	Bookshelf	0.64	1.35	3.54	3.70	2.19
People	Minimal room	1.40	1.40	2.56	2.8	1.16
	Person walking	1.40	1.40	2.56	2.8	1.16

6.2 Classification Performance

The NB Gaussian classifier was evaluated in 9 different environments. A given test scenario for an experiment would serve to see the impact of some scenario on the classifier performance, namely *placement*, *obstacles & disturbances*, and *people*. The test scenario *placement* served to analyze the impact of the speakers' position with respect to one another. The envi-

ronments in this test scenario were *adjacent room closer*, *multiple floors*, and *multiple rooms*. Moreover, the test scenario *obstacles & disturbances* focused on analyzing the impact that different obstacles and interference, such as furniture and RF disturbances, have on the classifier's prediction performance. The environments that belonged to this test scenario were *bookshelf*, *plaster wall*, *concrete wall*, and *RF disturbance*. Finally, the last test scenario *people* focused on capturing the impact that people have on the classifier's performance. The environments in this test scenario were *minimal room* and *person walking*.

Figure 6.3, Table 6.2, and Table 6.3 depict the classifier's performance evaluation and highlight that the classifier performed successfully above the AUC criteria in 7 out of the 9 environments. The corresponding environments were *minimal room*, *person walking*, *concrete wall*, *multiple floors*, *multiple rooms*, *plaster wall*, and *rf disturbance*. The previously mentioned environments will be further referred to as the *successful environments*. From the *successful environments* the environment *multiple rooms* resulted in the weakest performance, while the environment *person walking* resulted in the best performance, having an AUC of 86 % and 100 % respectively. A performance of 100 % is not surprising for the environment *person walking*, since this environment was one of the those that was used to train the classifier. On the other hand, the classifier did not meet the performance criteria in just 2 environments, namely *adjacent room closer* and *bookshelf*, having a performance of 37 % and 70 % respectively. The environments *adjacent room closer* and *bookshelf* will be referred to as the *failing environments*.

Table 6.2: Gaussian classifier's performance highlight in the different environments whereby the classifier failed to achieve the acceptable performance. The classifier used the 2 features MSUM and SSR. The metrics used in the comparison are the area under curve (AUC), log loss, and some terms from the confusion matrix, namely the terms false positive rate (FPR), false negative rate (FNR).

		Performance metric			
		AUC [%]	Confusion matrix		Log loss
Test scenario	Environment		FPR [%]	FNR[%]	
Placement	Adjacent room closer	36.99	42.00	84.00	4.12
Obstacles & disturbances	Bookshelf	70.00	60.00	0.00	1.33

The metrics AUC, FPR, FNR, and log loss will be used to clarify why the classifier did not perform well in failing environments.

Section 2.2 highlights that the AUC is a performance metric that compares the true positive rate (TPR) with the false positive rate (FPR) for different decision lines. Additionally, that the TPR and FPR are related to the *type*

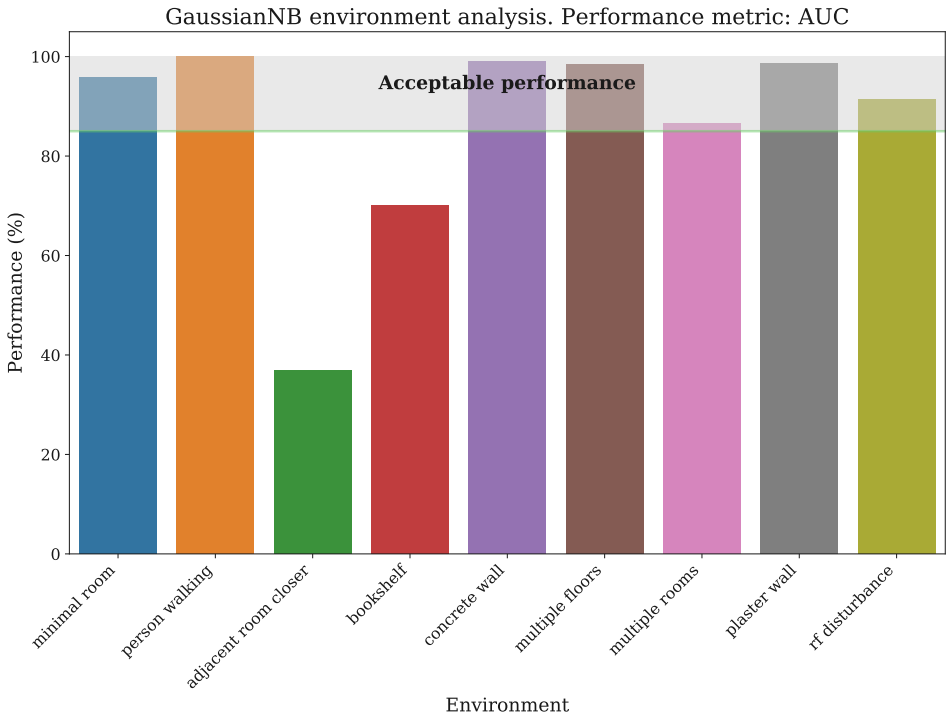


Figure 6.3: Gaussian Naive Bayes classifier's evaluation on several environments. Performance metric is area under curve (AUC). Accepted performance is 82% and greater. Only 2 out of 9 environments failed, namely adjacent room closer and bookshelf.

Table 6.3: Gaussian classifier's performance highlight in the different environments whereby the classifier succeeded above acceptable performance. The classifier used the 2 features MSUM and SSR. The metrics used in the comparison are the area under curve (AUC), log loss, and some terms from the confusion matrix, namely the terms false positive rate (FPR), false negative rate (FNR).

Test scenario	Environment	Performance metric			
		AUC [%]	Confusion matrix		Log loss
			FPR[%]	FNR[%]	
Placement	Multiple floors	98.51	3.00	0.00	0.31
	Multiple rooms	86.64	26.00	0.00	1.21
Obstacle & disturbances	Plaster wall	98.68	3.00	0.00	0.24
	Concrete wall	99.11	2.00	0.00	0.09
	RF disturbance	91.46	1.00	16.00	0.37
People	Minimal room	95.83	8.00	0.00	0.19
	Person walking	100.00	0.00	0.00	0.06

2 and *type 1 errors*, respectively. Furthermore, TPR is negatively related to the false negative rate (FNR).

In the *failing environments* the classifier inaccurately classified a significant amount of the observations that belonged to nodes in *different rooms* as being in the *same room*. Furthermore, Figures 6.7 and G.1 show that the observations' distribution for a given class has varied and spread towards the distribution of the next class label. The classifier achieved a false positive rate (FPR) of 42 % and 60 %, respectively the minimum and maximum AUC. Such FPR values indicate that the classifier have often made *type 1 errors*. In other words, the classifier inaccurately classified nodes to be in the *same room*, whereby this was done the most for the environment *bookshelf*.

The FNR was at its minimum 0 % and maximum 84 %, respectively for the environment *bookshelf* and *adjacent room closer*. For the latter environment, the classifier predicted almost all the observations from the class *same room* to be in the class *different rooms*.

Moreover, the classifier's log loss was at its minimum 1.33 and maximum 4.12, respectively for the environment *bookshelf* and *adjacent room closer*. According to Equation 2.6 the acceptable log loss is $\leq$1.08, since Table 5.5 highlights that the class prevalence was $\approx$34 % for the *same room*. The log loss results indicates that the classifier did not perform well in the environments *bookshelf* and *adjacent room closer*.

To the contrary to the *failing environments*, in the *successful environments* not much of the data distribution of a given class varied and spread out near the distribution of the other class.

A minimum FPR of 0 % and maximum FPR of 26 % was achieved, respectively for the environments *person walking* and *multiple rooms*. The NB Gaussian classifier barely made predictions that were a *false alarm*, in other words they were not much *type 1 errors*. In the *successful environments* the average and standard deviation of the false positive rate (FPR) was 6.14% and 9.12 % respectively. This is highlighted in Table 6.4, which summarizes the mean and standard deviation for the FPR, FNR, and log loss for the *successful environments*. Furthermore, Table 6.4 implies that the environment *multiple rooms* can be considered an edge case since it deviates from the mean FPR by $\approx$2.18 σ, comparing to the other environments, which deviated at their maximum within 1 σ.

All successful environments achieved an FNR of 0 % except the environment *rf disturbance*, which achieved a maximum FNR of 16 %. This indicates that the classifier was very specific in only responding when BLE nodes were in the *same room*. The average and standard deviation of the FNR for the successful environments were 2.29 % and 6.05 % respectively. The environment *rf disturbance* can be considered an edge case since it deviated by $\approx$2.27 σ, in comparison to the other environments that remained within 1 σ. Finally, the classifier had a minimum log loss of 0.06 and maxi-

mum log loss of 1.21, respectively for the environments *person walking* and *multiple rooms*. The environment *multiple rooms* exceeded the acceptable log loss of 1.08, yet the classifier succeeded in this environment.

In the previous discussion the environments *multiple rooms* and *rf disturbance* were considered edge cases. However, the classifier exceeded the acceptable AUC in those environments. Overall, this means that the TPR was more than the FPR.

Table 6.4: Highlight of the average and standard deviation classifier's errors amongst all of the successful environments. Metrics used are log loss and parts of the confusion matrix, namely, the false positive rate (FPR) and the false negative rate (FNR).

Description	FPR [%]	FNR [%]	Log loss
mean	6.14	2.29	0.35
standard deviation	9.12	6.05	0.39

The remainder of this section will compare the results of the *failing environments* in Table 6.2 to the average errors experienced in the *successful environments*, as it is highlighted in Table 6.4. When comparing the FPR, the environment *adjacent room closer* was ≈3.92 σ, while *bookshelf* was ≈5.91 σ. For the FNR the environment *adjacent room closer* deviated by ≈13.51 σ, while *bookshelf* was within 1 σ. The log loss was ≈9.56 σ for *adjacent room closer* and ≈2.58 σ for *bookshelf*. In the *failing environments* the classifier has always inaccurately classified a greater portion of the observations from nodes in *different rooms* to be in the *same room*. This can be seen for example in Figure G.1 from Section G. For only one of the *successful environments*, *multiple rooms*, did the classifier inaccurately classify observations from *different rooms* to be in the *same room*. However, for the latter environment the classifier still achieved a successful AUC.

6.3 Discussion

The NB Gaussian classifier made use of several features (predictors) in order to determine the probability that an observation belonged to a given class. This is shown in Equation 4.1. The first step in determining why the classifier failed in the environments *adjacent room closer* and *bookshelf* is, therefore, to determine if the NB Gaussian classifier model's assumptions were met. It was assumed that all features were statistically independent from one another and that the data distribution of the features, for each class, had a Gaussian distribution.

Independent features This section will focus first on the independence of the features. In none of the environments the data distribution showed

a strong statistical independence. Therefore, this is not the reason why the NB Gaussian classifier failed in certain rooms.

According to [44] if the predictors are independent then Equation 6.1 should always hold true for the conditional/posteriori probability. That is, the joint probability density function (pdf) for the posteriori probability is equal to the product of the pdf of all the posteriori probability of the individual features.

$$p(\mathbf{x}|w_i) = p(x_1|w_i)p(x_2|w_i), i = 1, 2 \tag{6.1}$$

Independent predictors/features will always have a covariance and correlation of 0. However, a covariance or correlation of 0 does not, in general, mean that the variables are statistically independent [44]. The correlation between the features *MSUM* and *SSR* for different environments are shown in Table 6.5 and 6.6, respectively for the environments where the Gaussian classifier did not and did exceed the accepted performance. The results serve as an indication for in which environment the two features may be independent.

The correlation is represented in the form of a correlation coefficient, which was calculated using the *Pearson* method. The coefficient range is [-1,1] and serves to show the linear dependency between variables. The linear dependency is weaker for values that are closer to 0. Moreover, a strong linear dependency is observed for a values closer to -1/1. Negative values highlight a negative relation, that is if one variable increases the other decreases, while in positive values both values move in the same direction. Note however, that the correlation coefficient highlights linear relationship and not the true feature dependent in the statistical form.

Table 6.5: Correlation of the features per class for the features MSUM and SSR accuracy for different environments where the Gaussian classifier did not pass the acceptable performance.

Test scenario	Environment	Correlation(MSUM,SSR)	
		Same room	Different rooms
Placement	Adjacent room closer	-0.440	-0.499
Obstacle & disturbances	Bookshelf	-0.391	-0.820

As the RF signals encounter variation overtime, due to obstacles or reflection, the feature *SSR* increases. Obstructions reduced the RSS and therefore the feature *MSUM* of the correlations are negative. Moreover, reflection may increase or decrease the signal strength. If the correlation is negative, it can imply that the RSS decreases as the variation increases, due to obstacles or reflection. In a barely furnished room, such as the environment *minimal room* and *person walking*, the correlation slightly went positive. This can be due to the fact that the table material reflects the signal such that the

Table 6.6: Correlation of the features per class for the features MSUM and SSR accuracy for different environments where the Gaussian classifier did pass the acceptable performance.

Test scenario	Environment	Correlation(MSUM,SSR)	
		Same room	Different rooms
Placement	Multiple floors	-0.453	-0.240
	Multiple rooms	-0.181	-0.465
Obstacle & disturbances	Plaster wall	-0.756	-0.160
	Concrete wall	-0.075	-0.512
	RF disturbance	-0.811	-0.349
People	Minimal room	-0.305	0.087
	Minimal room[1]	-0.420	-0.582
	Person walking	0.414	-0.777
	Person walking[1]	-0.164	-0.168

[1] *Dataset is the training data.*

signal strength increased. Another matter is the strong correlation for the environment *bookshelf*, *rf disturbance*, and *person walking*. In these environments the nodes which experience the most reflection experience also slight increase in the average sum of the RSS. This indicates that the reflection caused an amplification of the received signal [32].

From the group of environments where the classifier was not successful, not all environments showed a true statistical independence between features. In the environment bookshelf for example, the features were strongly linearly related for the samples from the class *other_room*.

In the set of environments where the classifier was successful, the features did not always have true linear independence. Although this assumption was not always met, the classifier succeeded in all the environments. Therefore, this assumption does not seem like the main factor why the classifier failed in those environments shown in Table 6.5.

Gaussian distribution The next assumption that was made for the Gaussian classifier was that the pdf class-conditional probability had a Gaussian distribution form, as it is shown in Equation 4.2. In other to validate this assumption a normality test was done using the python libraries *statsmodel* and *QQ-test*. A normality test is a test used to evaluate the if a distribution was probably sampled from a normal Gaussian distribution. Additionally, the Sci-kit learn library *shapiro* was used, which implements the Shapiro-Wilk test. The remainder of this section will focus on highlighting some of the results' main observations, which may clarify why the NB Gaussian classifier performed poorly in some of the environments and in others not.

Shapiro-Wilk Test The Shapiro-Wilk test is a statistical test to quantify how likely it is that a dataset was drawn from a normal Gaussian distribution dataset. Appendix E highlights the results. The results did not yield promising results, which can help distinguish between the results for failing and successful environments. In both successful and failing environments there are features that do not seem to be sampled from a Gaussian distribution [7].

This test returns two values, W and p. Additionally, a null hypothesis is assumed, which is that the sample was drawn from a Gaussian distribution. the p values are compared against a threshold alpha, typically 5 %. Values equal or less than alpha indicate that the hypothesis is false and must be rejected. That is, the sample most likely has not come from a Gaussian distribution. On the other hand, if p is larger than alpha, then it is not possible to reject the HO, so the sample may have probably come from a Gaussian distribution. Moreover, W values serves a similar purpose. Values closer to 1 indicates that the sample may have came from a Gaussian distribution, while lower values are not likely.

Table 6.7 and 6.8 shows the results for the Shapiro-Wilk test for a failing and successful environment, respectively. It can be seen that for both environments the p value rejects the H0, which means that is very unlikely that samples came from a Gaussian distribution.

In the environment *adjacent room* the W value indicates that it is unlikely that the feature *MSUM* for the *same room* may have been sampled from a Gaussian distribution. A successful environment such as *concrete wall* shows that it is unlikely that he feature *MSUM* for the class *same room* may have came from a Gaussian distribution. So not all features had a high W value.

This Shapiro-Wilk test does not seem to be a helpful test to clarify the failing behavior of the NB Gaussian classifier, because the W and p values are not completely distinguishable between failing and successful environments.

Quantile-Quantile (QQ)-test The QQ-test is a visual test in order to validate if two datasets came from populations that have a similar distribution. Appendix F highlights the results for the QQ-test. The results highlight that in the failing environments most of the features' distribution for the different class labels were not likely to be Gaussian or most of the observations from a feature did not resemble a Gaussian shape or both. In the environments where the classifier was successful, this was not true [7].

For this analysis the dataset is compared against a Gaussian distribution. A dataset that may have a Gaussian distribution would have a QQ-curve that is linear on a 45 degree angle, which represent the curve for an ideal Gaussian distribution with a similar distribution, after comparing the quantiles of each dataset.

Table 6.7: Shapiro-Wilk test for a failing environment adjacent room closer. W values close to 1 or p values > indicate that the feature's distribution most probably was sampled from a Gaussian distribution. Otherwise, low W values or p values ≤indicate that the feature's distribution most probably was not sampled from a Gaussian distribution.

Class label	**Feature**	**W**	**p**
Same room	SSR	0.86	$2.41 * 10^{-5}$
	MSUM	0.55	$2.67 * 10^{-11}$
Different rooms	SSR	0.78	$3.05 * 10^{-16}$
	MSUM	0.91	$1.58 * 10^{-10}$

Table 6.8: Shapiro-Wilk test for a successful environment concrete wall.W values close to 1 or p values > indicate that the feature's distribution most probably was sampled from a Gaussian distribution. Otherwise, low W values or p values ≤indicate that the feature's distribution most probably was not sampled from a Gaussian distribution.

Class label	**Feature**	**W**	**p**
Same room	SSR	0.91	$1.30 * 10^{-3}$
	MSUM	0.34	$8.64 * 10^{-14}$
Different rooms	SSR	0.95	$1.47 * 10^{-5}$
	MSUM	0.84	$2.87 * 10^{-12}$

Figure 6.4 depicts the results for one of the environments where the classifier did not meet the acceptable performance, namely *adjacent room closer*. It can be seen that for no given class label all variables are following the ideal curve. For speakers in the same room, the feature *SSR* has a strong possibility for having a Gaussian distribution. However, the feature *MSUM* seems not have a Gaussian distribution at all. This holds true for the dataset from the other class label $different\_rooms$.

QQ-plot comparison. Environment: adjacent room closer

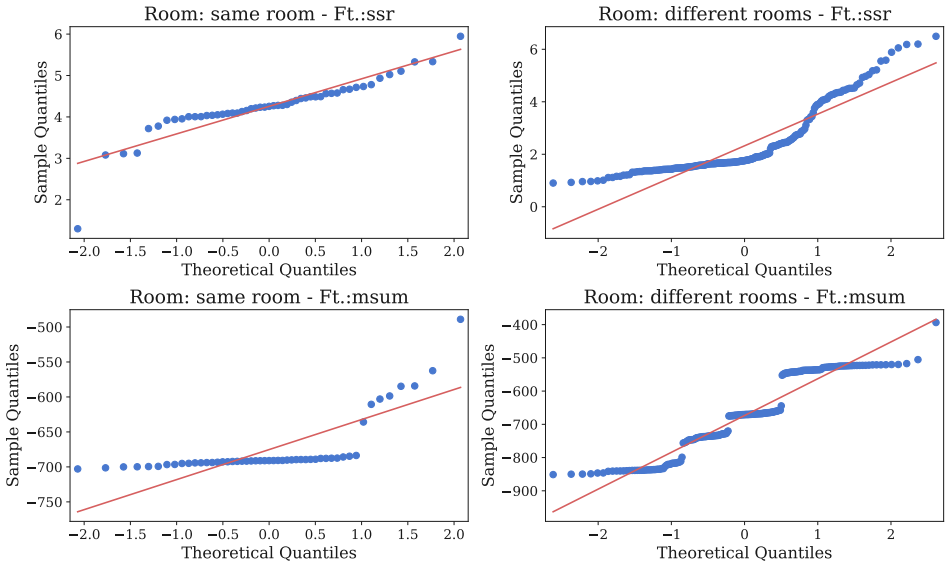


Figure 6.4: QQ-test for failing room: adjacent room environment

On the other hand, the behavior is different for an environment in which the classifier performed well. See the environment *concrete walls* in Figure 6.5. The data distribution for the the feature *SSR* seems to have a Gaussian distribution. However, this is not so much the case for the feature *MSUM*, especially for the class *same room*. The feature *MSUM* in the class *different rooms* follows the ideal curve most of times, indicating that most part of the data distribution resembles samples from Gaussian distribution.

All environments both successful or not, have encountered a scenario where the distribution for a feature for a given class is not an ideal Gaussian distribution. However, there are some patterns that can be observed. First, in almost all cases where a feature has not been considered Gaussian it has occurred at least for the feature *MSUM* and mainly in the class *same_room*. Second, in the environments where the classifier performed poorly, most of the data distributions were not Gaussian or most of the quantiles for a single given feature in a class label were not following the line or both. See Figure 6.4 for example, the feature *SSR* seems to follow the ideal line a bit for the

class *same room*, but not much for the class *different rooms*. The same goes for the feature *MSUM*, for the class label *same room* barely follows the line, and deviates for most of the quantiles. For the class label *different rooms* the quantiles partly follow the line.

To the contrary to the challenging environments, in the environments where the classifier performed successfully, the QQ-curve followed the ideal line for most of the features. In Figure 6.5, most of the data samples for all but one distribution followed the ideal curve. Only the distribution for the feature *MSUM* for the class *same_room* barely followed the line. There were more cases where the distribution did not follow the ideal line well. Consider for example the environment *minimal_room* in Figure F.4. However, there were not much data samples and there were very minimal outliers and most of the data points followed the line. In the environment *multiple_rooms* the distribution for the feature *MSUM* in *same_room* seemed not to be completely Gaussian, however most of the other distributions were Gaussian enough. The same goes for the environment $rf\_disturbance$. Therefore, the amount of data points from the quantiles that follow the ideal Gaussian line versus those that did not seem to play an important role for the this Gaussian classifier.

The assumption is that the classifier failed in the given environments because in most the distribution there was not a proper Gaussian distribution.

QQ-plot comparison. Environment: concrete wall

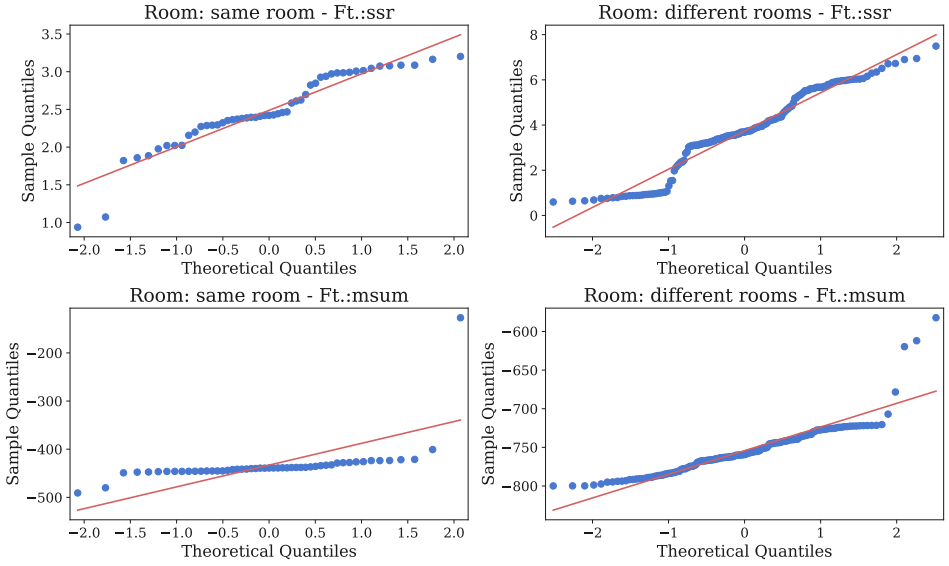


Figure 6.5: QQ-test for successful room:concrete wall

Data Distribution of the Features In the environments where the classifier failed the signal strength of the nodes from different rooms become stronger on the average, beyond the average signal strength for the dataset used in the training phase. This occurred due to the fact that the distance between nodes from different room became smaller.

The AUC will be low for having a low true positive rate (TPR) or false positive rate (FPR). Table 6.2 and 6.3 highlight that the 3 environments with a low TPR (high FNR) and a high FPR were the environments *adjacent room closer*, *bookshelf*, and *multiple rooms*. Only the latter environment achieved an AUC that was acceptable.

Figure 6.6 shows the distribution of the feature *MSUM* for both classes. In order for the false positive rate to increase, the classifier had to inaccurately classify observations from nodes in *different rooms* as being in the *same room*. The features' box plots highlights that the Interquartile Range (IQR) of almost all environments are separable, except that of *adjacent room closer*. It can be seen that the IQR of both classes overlap. This can also be seen in Figure 6.7. Appendix G highlights how the classifier classified observations in the remaining environments. As the AUC is being calculated and the decision line moved, a prediction occurs where FPR 42 % and the TPR 16 % is achieved. This brings the AUC to be too low, such that the prediction are a reciprocal. That is, the observation from one class is classified to be in the wrong class. This occurred because the feature *MSUM* for nodes in the *same room* have gotten weaker. This is probably caused by interference from the BLE node that was in very close proximity. Moreover, the feature *MSUM* for the class *different rooms* has gotten stronger, since the nodes from different rooms are really close. According to Table 6.1 the minimum distance between nodes from the *different rooms* was 0.5 meters, making the signal much stronger.

The environment *bookshelf* had the IQR also separable for both classes. However, this environment failed, but by missing an AUC of 10 %. The environment *multiple room* had also a relatively high FPR, but the classifier was successful. The difference between the environments *bookshelf* and *multiple rooms* is the FPR value. In other words, the ratio of observations from nodes in *different rooms* that were inaccurately classified as in the *same room*. However, the environment *bookshelf* had its IQR values closer to the decision lines, indicating that more observations were inaccurately classified. This increase in the feature *MSUM* indicates that over time the signal from nodes in *different rooms* became stronger than usual. According to Table 6.1 the minimum distance between nodes from *different rooms* for the environment *bookshelf* was 3.54 meters, while for the environment *multiple rooms* it was 3.70 meters. These distance indicates that for the environment *bookshelf* nodes were closer to one another, thus the signal will be slightly stronger.

The *successful environments* have in common that their IQR of distri-

bution for the feature *MSUM* for a given class does not overlap with the other class. Additionally, The IQR of a feature does exceed far beyond the decision line.

On the other hand, the distribution of the feature *SSR* was not distinguishable between the *failing and successful environments*, as Figure H.1 shows. So the feature *MSUM* is a factor that plays a role when a classifier fails, and not the feature *SSR*.

In conclusion both environments *bookshelf* and *adjacent room closer* failed because the data distribution for the feature *MSUM* has overlapped beyond the optimized decision line for the class *different rooms*. In other words, the classifier have often inaccurately classified observations belonging to nodes from *different rooms* as belonging to nodes in the *same room*. This occurred mainly when the average of the sum of the signal strength became strong over time. A scenario that caused such an effect is when the minimum distance between nodes in *different rooms* were less than 1 meter. Additionally, when nodes were placed in an area that was extremely occupied, such as a bookshelf.

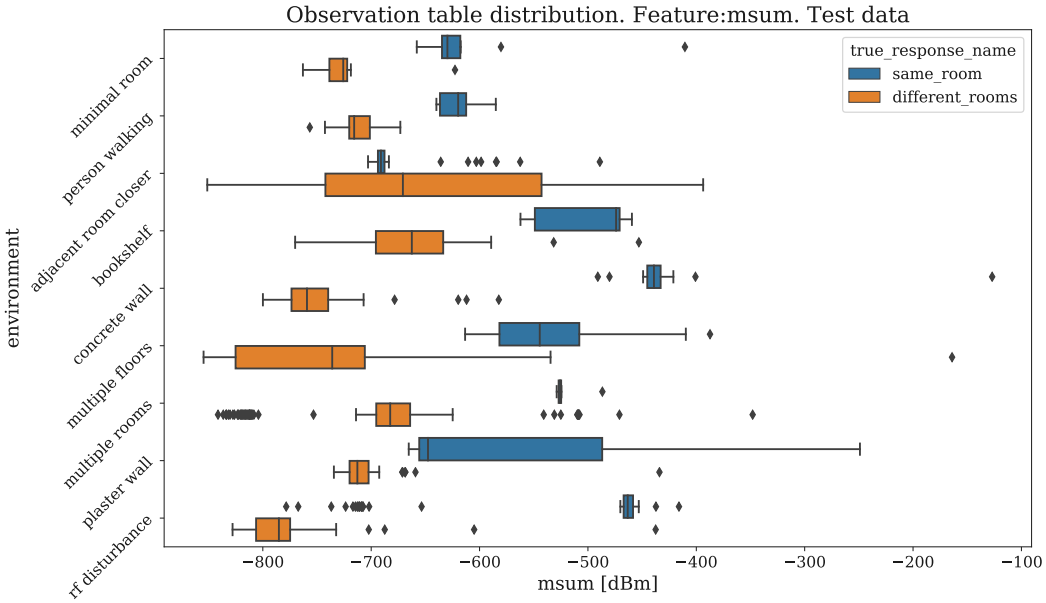


Figure 6.6: Box plot describing the distribution of the feature MSUM. For every environment the distribution is compared for both classes, same room versus different rooms

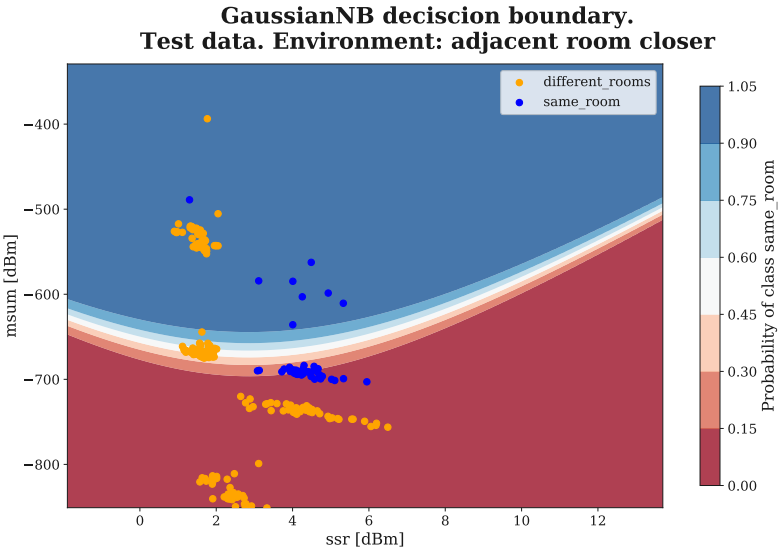


Figure 6.7: NB Gaussian decision line and scatter plot for the environment adjacent room closer. A scatter plot is shown for each observation from the test data that belongs to either of the two classes, same room or different rooms. Every observation is given a probability of the likelihood of it belonging to the positive class, same room

This page was deliberately left blank

Chapter 7

Conclusion and Future Work

This chapter elaborates on the key findings and conclusion in this thesis. Furthermore, future work will be highlighted on other improvements that can be done.

7.1 Conclusion

For Follow Me, which is being developed by Kien, it is required for the Bluetooh Low Energy (BLE) speakers to be categorized based on which are in the same room together. BLE speakers will be referred to as BLE nodes. Indoor localization methods, which will be used for Follow Me, suffer from several challenges. First, BLE signals may suffer from interference indoors from other devices on the 2.4 GHz such as a microwave. Additionally, the signals may diffract, scatter, reflect, and may be obstructed by obstacles in a room such as walls, furniture, and people. Consequently, this causes the signal to experience more distortion and variation. Second, it is difficult to model the radio frequency (RF) propagation for all environments, since every environment is occupied differently with furniture and people. Third, the environment indoors changes frequently making it difficult for some localization methods to maintain their accuracy. We therefore, researched how to accurately group BLE nodes by the room they are located, while being robust to radio-frequency (RF) interference, attenuation, and the multipath effect caused by people, furniture, and obstacles.

We have developed a Naive Bayes (NB) Gaussian classifier that uses only two features. Namely, the *Mean of the set of RSS chunks' Sum (MSUM)* and the *Standard deviation of the set of RSS chunks' Range (SRR)*. The classifier was designed to distinguish between observations from a pair of nodes that are in the *same room* and in *different rooms*. We have evaluated the classifier in 9 environments, whereby a subset of the samples from only 2 environments were used for training the classifier. The classifier achieved an area under the curve (AUC) $\geq$80 % in 7 out of 9 environments, thus even

in unseen environments. The classifier's performance meets Kien desires to have a method that is robust indoors and is able to function well in unseen environments.

Our method is similar to fingerprint-based method since both methods analyze the RSS signal from the environment, making no distinction between line-of-sight (LOS) and non-LOS scenarios. Taking LOS and non-LOS scenarios into account is a property that makes for more robust methods compared to others such as triangulation, trilateration, and multilateration. However, fingerprint-based methods require that data be collected and analyzed in all of the locations where localization is desired. Additionally, mainly the RSS is used. Our method differs in that it uses different features, which are generated from the RSS information. Additionally, our method performs well in completely unseen environments. Our method also showed to work in an environment with RF interference.

However, our method has limitations. The classifier mainly fails in environments where the features' distribution deviates from having a Gaussian-like shape. Such a scenario can occur when nodes from *different rooms* are within 1 meter of each other, while being further away from nodes in the *same room*. Additionally, when nodes are obstructed by a significant amount of obstacles, which may negatively impact most of the signals.

7.2 Future Work

The feature *MSUM* gave the best discriminative information such as to distinguish between observations belonging to a pair of nodes from the *same room* and *different rooms*. However, the feature *MSUM* has its limitations.

Therefore, more features should be found that can be robust to changes in RF signal. With more features, the model should be retrained, potentially yielding a better model.

The next step towards the Follow Me feature is taking the classifier output and deducing the different groups exist and categorizing all speakers to the groups, also known as the *list of bundles*.

Bibliography

[1] Average Household Size in the Netherlands.

[2] Central Limit Theorem Explained - Statistics By Jim.

[3] classification - Machine learning classifiers big-O or complexity - Cross Validated.

[4] Computational Complexity of ML Models - Paritosh Kumar - Medium.

[5] Dutch statistics & Facts about the Netherlands.

[6] Feature Selection in Python with Scikit-Learn.

[7] A Gentle Introduction to Normality Tests in Python.

[8] GitHub - AltBeacon/spec: AltBeacon Technical Specification.

[9] An Implementation and Explanation of the Random Forest in Python.

[10] kien | Kien: portable multiroom sound system, and makes your music follow you.

[11] machine learning - What is the O() runtime complexity of AdaBoost? - Stack Overflow.

[12] Triangulation, Trilateration, or Multilateration? | Circuit Cellar.

[13] Understanding AUC - ROC Curve - Towards Data Science.

[14] What is the time complexity of a Random Forest, both building the model and classification? - Quora.

[15] Whats considered a good Log Loss in Machine Learning ?

[16] Yugo Agata, Jihoon Hong, and Tomoaki Ohtsuki. Room-level proximity detection using beacon frame from multiple access points. In *2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, pages 941–945, Hong Kong, Dec. 2015. IEEE.

[17] Robert Akl, Karthik Pasupathy, and Mohamad Haidar. Anchor nodes placement for effective passive localization. In *2011 International Conference on Selected Topics in Mobile and Wireless Networking (iCOST)*, pages 127–132, Shanghai, China, Oct. 2011. IEEE.

[18] Jae Hyung An and Lynn Choi. Inverse fingerprinting: Server side indoor localization with Bluetooth low energy. In *2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pages 1–6, Valencia, Spain, Sept. 2016. IEEE.

[19] Sinem Bozkurt, Gulin Elibol, Serkan Gunal, and Ugur Yayan. A comparative study on machine learning algorithms for indoor positioning. In *2015 International Symposium on Innovations in Intelligent SysTems and Applications (INISTA)*, pages 1–8, Madrid, Spain, Sept. 2015. IEEE.

[20] Danijel Cabarkapa, Ivana Grujic, and Petar Pavlovic. Comparative analysis of the Bluetooth Low-Energy indoor positioning systems. In *2015 12th International Conference on Telecommunication in Modern Satellite, Cable and Broadcasting Services (TELSIKS)*, pages 76–79, Nis, Serbia, Oct. 2015. IEEE.

[21] Stijn Crul, Geoffrey Ottoy, and Lieven De Strycker. Location awareness enables autonomous commissioning in wireless sensor networks. In *2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–6, Alcala de Henares, Spain, Oct. 2016. IEEE.

[22] Hui Dai and Richard Han. TSync: a lightweight bidirectional time synchronization service for wireless sensor networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 8(1):125, Jan. 2004.

[23] Salim el Khediri, Nasri Nejah, Abdennaceur Kachouri, and Anne Wei. Survey of synchronization algorithms: Wireless sensor networks case study. *The 2nd International Conference on Communications and Information Technology (ICCIT): Wireless Communications and Signal Processing, Hammamet*, 06 2012.

[24] Jeremy Elson, Lewis Girod, and Deborah Estrin. Fine-Grained Network Time Synchronization using Reference Broadcasts. *ACM SIGOPS Operating Systems Review*, page 14, Dec. 2002.

[25] Saurabh Ganeriwal, Ram Kumar, and Mani B Srivastava. Timing-sync Protocol for Sensor Networks. *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 138–149, Nov. 2003.

[26] Yu Gu and Fuji Ren. Energy-Efficient Indoor Localization of Smart Hand-Held Devices Using Bluetooth. *IEEE Access*, 3:1450–1461, 2015.

[27] Goran Horvat, Snjeana Rimac-Drlje, and Drago agar. Fade Depth Prediction Using Human Presence for Real Life WSN Deployment. *RADIOENGINEERING*, 22(3):11, 2013.

[28] Myungin Ji, Jooyoung Kim, Juil Jeon, and Youngsu Cho. Analysis of positioning accuracy corresponding to the number of BLE beacons in indoor positioning system. In *2015 17th International Conference on Advanced Communication Technology (ICACT)*, pages 92–95, Phoenix Park, PyeongChang, South Korea, July 2015. IEEE.

[29] Zhu Jianyong, Luo Haiyong, Chen Zili, and Li Zhaohui. RSSI based Bluetooth low energy indoor positioning. In *2014 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 526–533, Busan, South Korea, Oct. 2014. IEEE.

[30] Salim el khediri. Analysis Study Of Time Synchronization Protocols In Wireless Sensor Networks. *International Journal of Distributed and Parallel systems*, 3(3):155–165, May 2012.

[31] Michal Kochl, Juraj Miek, and Peter evk. 2.4GHz ISM Band Radio Frequency Signal Indoor Propagation. pages 1027–1034. IEEE, Sept. 2014.

[32] Tarmo Koppel, Andrei Shishkin, Heldur Haldre, Nikolajs Toropovs, Inese Vilcane, and Piia Tint. Reflection and Transmission Properties of Common Construction Materials at 2.4 GHz Frequency. *Energy Procedia*, 113:158–165, May 2017.

[33] Srdjan Krco. Bluetooth Based Wireless Sensor Networks Implementation Issues and Solutions. *10th Telecommunications Forum (TELEFOR2002)*, page 7, 2002.

[34] Pavel Kriz, Filip Maly, and Tomas Kozel. Improving Indoor Localization Using Bluetooth Low Energy Beacons. *Mobile Information Systems*, 2016:1–11, 2016.

[35] Pius W Q Lee, Winston K G Seah, Hwee-Pink Tan, and Zexi Yao. Wireless sensing without sensorsan experimental study of motion/intrusion detection using RF irregularity. *Measurement Science and Technology*, 21(12):124007, Dec. 2010.

[36] Gilles Louppe. Understanding Random Forests: From Theory to Practice. *arXiv:1407.7502 [stat]*, June 2015. arXiv: 1407.7502.

[37] Miklós Maróti, Branislav Kusy, Gyula Simon, and Ákos Lédeczi. The flooding time synchronization protocol. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, SenSys 04, page 3949, New York, NY, USA, 2004. Association for Computing Machinery.

[38] Amit Nayyer. A Comparative study of Time Synchronization Protocols in Wireless Sensor Network. *International Journal of Computer Applications*, 36:7, Dec. 2011.

[39] Sepideh Nazemi Gelyan, Arash Nasiri Eghbali, Laleh Roustapoor, Seyed Amir Yahyavi Firouz Abadi, and Mehdi Dehghan. SLTP: Scalable Lightweight Time Synchronization Protocol for Wireless Sensor Network. In Hongke Zhang, Stephan Olariu, Jiannong Cao, and David B. Johnson, editors, *Mobile Ad-Hoc and Sensor Networks*, volume 4864, pages 536–547. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007. Series Title: Lecture Notes in Computer Science.

[40] Fernando Seco, Antonio R. Jimenez, Carlos Prieto, Javier Roa, and Katerina Koutsou. A survey of mathematical methods for indoor localization. In *2009 IEEE International Symposium on Intelligent Signal Processing*, pages 9–14, Budapest, Hungary, Aug. 2009. IEEE.

[41] M.L. Sichitiu and C. Veerarittiphan. Simple, accurate time synchronization for wireless sensor networks. In *2003 IEEE Wireless Communications and Networking, 2003. WCNC 2003.*, volume 2, pages 1266–1273, New Orleans, LA, USA, 2003. IEEE.

[42] Philipp Sommer and Roger Wattenhofer. Gradient Clock Synchronization in Wireless Sensor Networks. *2009 International Conference on Information Processing in Sensor Networks*, page 12, Aug. 2009.

[43] Bharath Sundararaman, Ugo Buy, and Ajay D. Kshemkalyani. Clock synchronization for wireless sensor networks: a survey. *Ad Hoc Networks*, 3(3):281–323, May 2005.

[44] Sergios Theodoridis and Konstantinos Koutroumbas. *Pattern recognition*. Elsevier Acad. Press, Amsterdam, 4. ed edition, 2009. OCLC: 550588366.

[45] Jana van Greunen and Jan Rabaey. Lightweight Time Synchronization for Sensor Networks. *WSNA '03: Proceedings of the 2nd ACM international conference on Wireless sensor networks and applications*, pages 11–19, Sept. 2003.

[46] Jiang Xiao, Zimu Zhou, Youwen Yi, and Lionel M. Ni. A Survey on Wireless Indoor Localization from the Device Perspective. *ACM Computing Surveys*, 49(2):1–31, June 2016.

[47] Ali Yassin, Youssef Nasser, Mariette Awad, Ahmed Al-Dubai, Ran Liu, Chau Yuen, Ronald Raulefs, and Elias Aboutanios. Recent Advances in Indoor Localization: A Survey on Theoretical Approaches and Applications. *IEEE Communications Surveys & Tutorials*, 19(2):1327–1346, 2017.

[48] Faheem Zafari, Athanasios Gkelias, and Kin K. Leung. A Survey of Indoor Localization Systems and Technologies. *IEEE Communications Surveys & Tutorials*, 21(3):2568–2599, 2019.

This page was deliberately left blank

Appendix A

Kien's Follow Me Feature

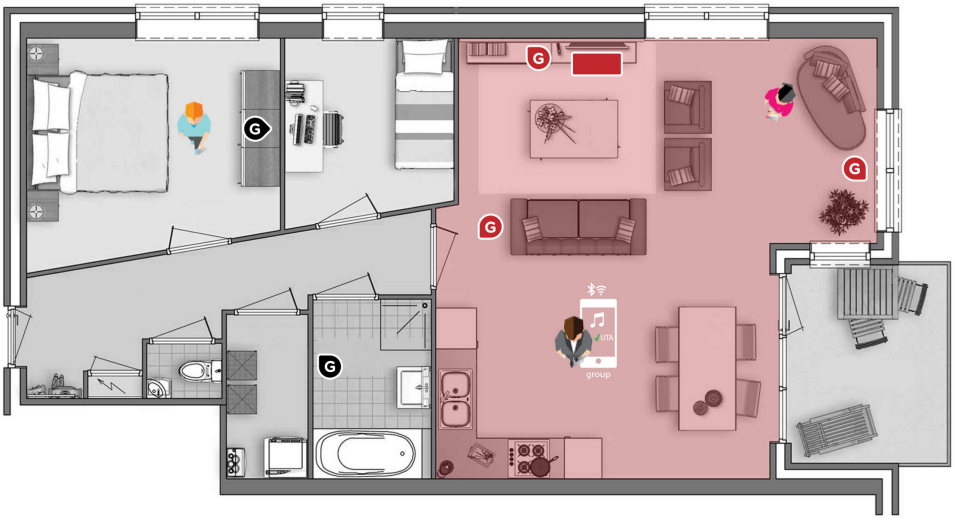


Figure A.1: Kien's 5.1 system with Follow-me activated in the user's smartphone with the Kien application. Audio is playing only through the speakers, which are shown in red, that are within the subwoofer's group (G) and located in the same room as the user carrying the smartphone with the Kien application [10].

The living room is highlighted red depicting that the user has Follow-me enabled on the mobile phone and music is currently playing only in that room. It is expected that if a user enters a room that does not contain any speakers then no speakers from the network should be playing music. Therefore, it is necessary to have a system that mainly detects when a user is in a room containing speakers that are in the audio network, and responds within a few seconds by playing music in that room.

Follow-me requires two stages. First, the system should know what speakers are together in the same room. Figure A.1 shows that there are 3 groups. Namely, 1 speaker in the living room, 1 speaker in the bathroom,

and 4 speakers in the living room. Secondly, the system must detect when the user, which is carrying the a smartphone, enters a room containing speakers. Therefore, the user position must also be deduced.

Appendix B

Model's Evaluation Flowchart

A classifier sometimes have hyper-parameters, which are parameters that are specific to that classifier and used for tweaking the its behavior. GridSearch is a Scikit library that makes use of cross validation to optimize a classifier. A classifier is evaluated using a performance metric CV score, whereby the higher the CV score the better the classifier's performance. We use of GridSearch with the training and testing observation table, which are the dataset *train_ot* and *test_ot*, respectively.

The Scikit library GridSearch was used to optimize a classifier in order to find the hyper parameter that resulted in the best performance given a dataset as input. The GridSearch makes use of cross validation. The dataset *train_ot* and *test_ot* were used. Cross validation was then applied on the training data, whereby k-1 fold was used for training and 1 fold used for validating. For a given classifier this process was repeated for different hyper-parameter combination. The performance, the CV score, of all these classifiers were evaluated and the best classifier containing the highest CV score was selected as the final classifier. This is done for for each classifier type. The final optimized models were then tested against the test data, which was a completely unseen dataset. The classifier's evaluation A, as shown in Figure B.1 , were compared and the best model was selected based on the highest *auc_roc* value. The procedure is as follows:

1. Parse the dataset for each environment;
2. Preprocess the dataset. Clean data, remove outliers etc.;
3. Merge dataset into the total dataset *unified_ds*;
4. Split the total dataset into two datasets, one for training the classifiers and evaluating which classifier is the best. Additionally, a next

dataset to evaluate the final classifier across different environments. Respectively, *train_test_ds* and *test_only_ds*;

5. Create an observation table from datasets. Given a set of samples generate an observation with all the features, which were selected in Section 4.1;

6. Split the *train_test_ot* into two datasets to train and test data, with a 70 % and 30 % split respectively;

7. Using the training data, *train_ot* , apply cross validation using K-Folds, with K=5. The training data is split in k folds. For each iteration the model is trained using a new set of k-1 folds of data and tested on the remaining 1 fold. At the end of each iteration a CV score metric will be given to that model. At the end of k iteration, average all the CV

8. Repeat with other model parameters;

9. Find optimal parameters that gives the highest average CV score;

10. Repeat for other classifiers;

11. For each classifier select the model with the optimal parameters. For each optimized classifier test against the test data, *test_ot*;

12. Select the classifier that has the best performance on the testing data, from **Evaluation A**;

13. Evaluate the final model also with test data from completely environment, namely *test_only_ot* resulting in **Evaluation B**.

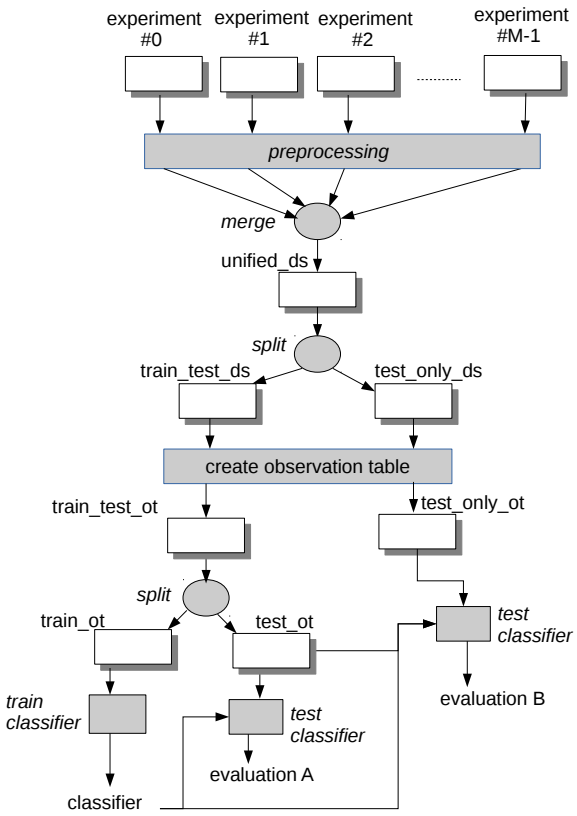


Figure B.1: Flowchart on how the dataset for $\boldsymbol{M}$ *different environments are merged together, preprocessed, and divided into two datasets. First the training and testing dataset, train_test_ds, to select final model. Second, test only dataset, test_only_ds, to evaluate final model in different environments.*

This page was deliberately left blank

Appendix C

BLE Packets

C.1 SLTP Advertisement Packet Structure

Table C.1: Data structure for the advertised Scalable Lightweight Time Synchronization Protocol (SLTP) packets. The data structure is not a standard form, but was implemented using the Altbeacon standard as reference. The beacon code make this packet distinguishable from other advertised BLE packets. The speaker ID is used to uniquely identify the transmitter of the packet. The elapsed time is transmitter elapsed time and it is used for synchronizing.

Bytes	Field name	Description	Value
2	Beacon code	Field used to distinguish the packet from other advertised packets	0xceed
1	Speaker ID	ID of the node that transmits the SLTP packet	[1]
2	Elapsed time	Local elapsed time of the transmitter of the SLTP packet	[1]

[1] *Calculated during runtime.*

C.2 Altbeacon Advertisement Packet Structure

Table C.2: Data structure for the advertised Altbeacon packet. The beacon code makes this packet distinguishable from other advertised BLE packets. The beacon ID consists of the company universally unique identifier (UUID), speaker ID, and the room ID [8].

Bytes	Field	Description	Value
2	Beacon code	Field to distinguish the packet type	0xbeac
16	Company UUID	ID to identify the owner of the Application Set to a custom Kien UUID	{0x01,0x02,0x03,0x04, 0x05,0x06,0x07,0x08, 0x09,0x0a,0x0b,0x0c, 0x0d,0x0e,0x0f, 0x10}
1	Speaker ID	ID of the transmitter of the Altbeacon packet elapsed time of the transmitter of the SLTP packet	[1]
1	Room ID	Estimated room the transmitter of this Altbeacon packet is in.	[1]

[1] *Calculated during runtime.*

Appendix D

Experiment Environments

It is essential to model the large variation of different speaker placement and home structure in order to have a classifier that covers most scenarios. The goal was therefore to model the edge cases and the most general home structure and placement. Homes in the Netherlands consist of a family of 2 on the average [1] [5]. Homes typically are one floor apartment with less than eight rooms. The scope of this thesis is therefore a home with no more than 4 people and a home with maximum eight rooms.

Each environment can be grouped in a test group that defines the purpose of the experiment in that environment. The test groups were *placement*, *obstacles & disturbances*, *people*. The remaining of the section will focus to describe the environment for each test group and the environments' setup.

It is interested to know the impact that people, obstacles and disturbances, and placement of the speakers have on the performance of the model. Each experiment were executed for approximately 3-5 minutes and done between four to five times.

D.1 Placement

The goal for the *placement* test case was to analyze the impact that the distance between the speakers on the classifier model. Additionally, the impact of the speaker density has on the model. During these experiments the altitude of the speaker and also the edge case, where a speakers from an adjacent rooms are closer to one another than speakers in the same room, were analyzed.

D.1.1 Adjacent Room Closer

This experiment was an edge case that was the main focus of this thesis research. BLE RSS is impacted by the distance between sender and receiver. Additionally, the obstacles that are between the RF path of two speakers

also influence the RF signals strength. Due to this fact two speakers that are in adjacent rooms, but relatively closer to one another despite being obstructed by a wall may have a lower signal strength that two speakers that are in the same room. This can happen if the speakers that are in the same room are far apart or, the disturbance in between both speakers are strong. If the RF signal strength is solely used for the classifier, speakers in *different rooms* may be perceived to be in the same room. The setup for this experiment is shown in Figure D.1.

Table D.1 shows the packet ratio for this experiment before applying data munging. 18 % of the data samples were from the class *same room*, and the remaining 82 % for *different rooms*. This dataset is highly imbalanced.

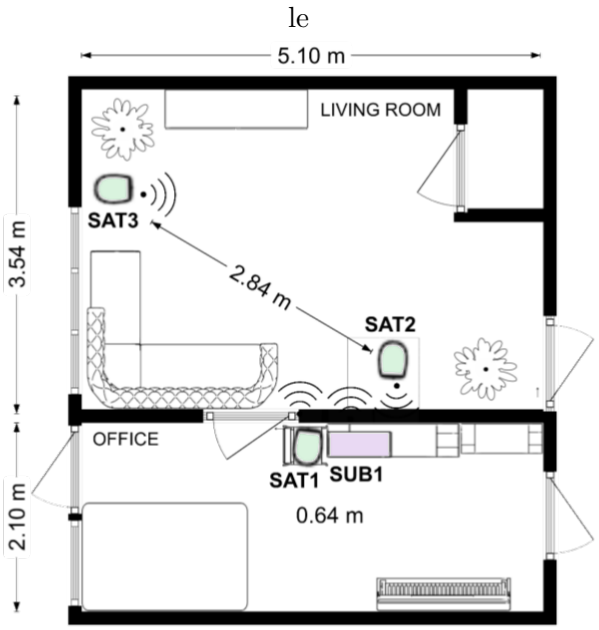


Figure D.1: Environment setup for the environment adjacent room closer. SAT2 is closer to the other speakers from the office than SAT3, which is from the same room.

Table D.1: Packet count ratio for all the classes in the environment adjacent room closer.

Class label	Amount
Same room	5784
Different rooms	26196
Total	31980

D.1.2 Multiple Rooms

The environment *multiple rooms* simulates an a usae case where a user places BLE nodes in multiple rooms. The setup for this experiment is shown in Figure D.2. Table D.2 shows the packet ratio for this experiment. Only 3 % of the dataset has been for the class *same class*. Although, this experiment was not the main scope of this research it has been done as an edge case experiment and to analyze how the system will behave.

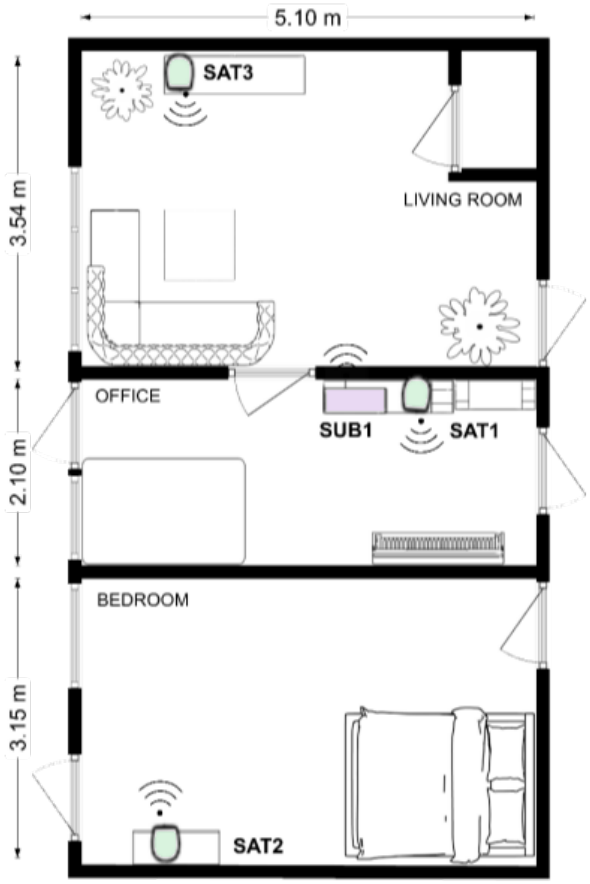


Figure D.2: Multiple rooms environment

D.1.3 Multiple Floors

Multiple floors is a use case outside of the scope of this research, however it was still taking into account to have an idea how the system will be have

Table D.2: Packet count ratio for all the classes in the environment multiple rooms.

Class label	Amount
Same room	1192
Different rooms	30788
Total	31980

in a home with multiple floors. Figure D.3 shows setup for this experiment and the packet ratio is shown in Table D.3. Only 23 % of the dataset was gathered for the same room.

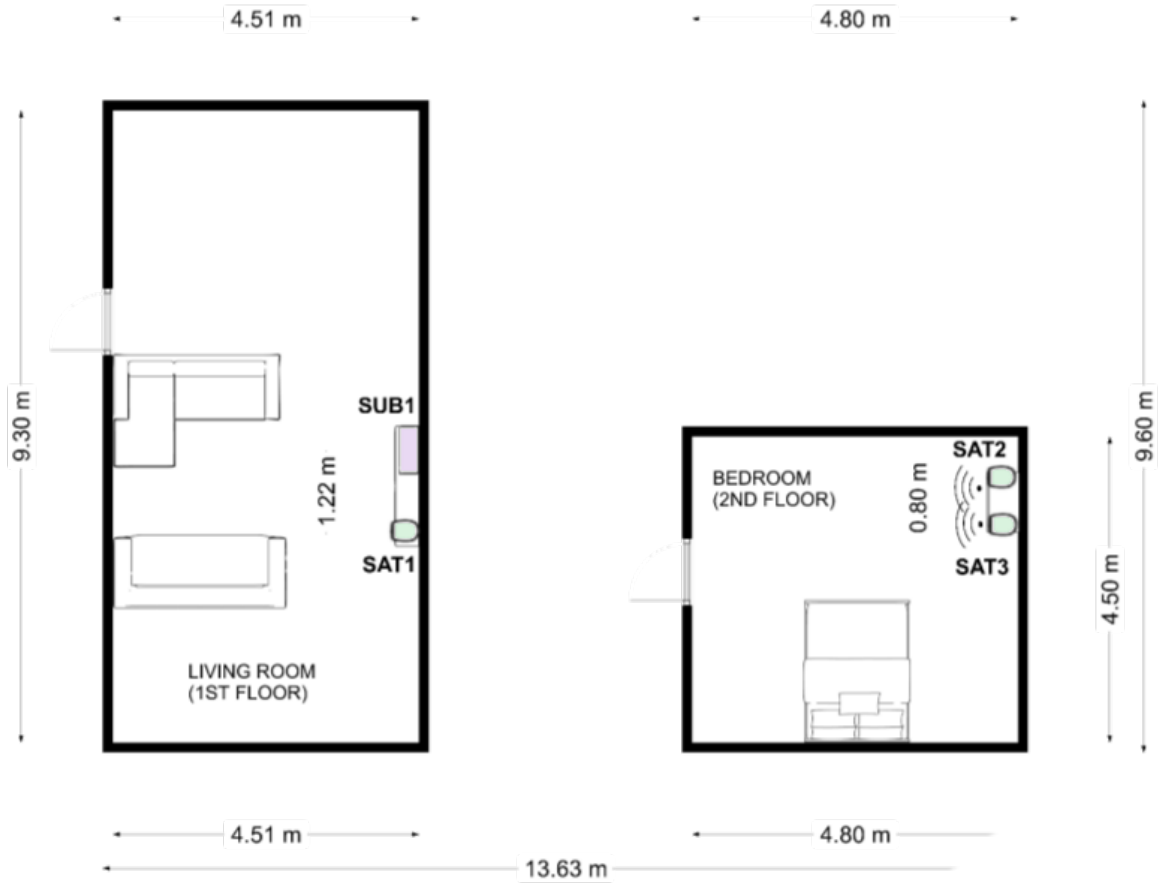


Figure D.3: Multiple floors environment

Table D.3: Packet count ratio for all the classes in the environment multiple floors.

Class label	Amount
Same room	5922
Different rooms	19662
Total	25584

D.2 Obstacles and Disturbances

Indoor environment consists of obstacles that is unique in every new environment. This include furniture and walls. The goal for experiments under this branch is to see how the type of obstacle impacts model. The experiments done were:

D.2.1 Plaster Wall

This experiment focused on analyzing the impact of a wall material on the model. The setup for this experiment is shown in Figure D.4. Table D.4 illustrates that 30 % of the data was gathered for the class same room.

Table D.4: Packet count ratio for all the classes in the environment plaster walls.

Class label	Amount
Same room	7771
Different rooms	17813
Total	25584

D.2.2 Concrete Wall

This experiment focused on analyzing the impact of a wall material on the model. The setup for this experiment is shown in Figure D.5. Table D.5 shows that 77 % of the dataset was for the class *different rooms*, making this experiment slightly imbalanced.

Table D.5: Packet count ratio for all the classes in the environment concrete walls.

Class label	Amount
Same room	5889
Different rooms	19695
Total	25584

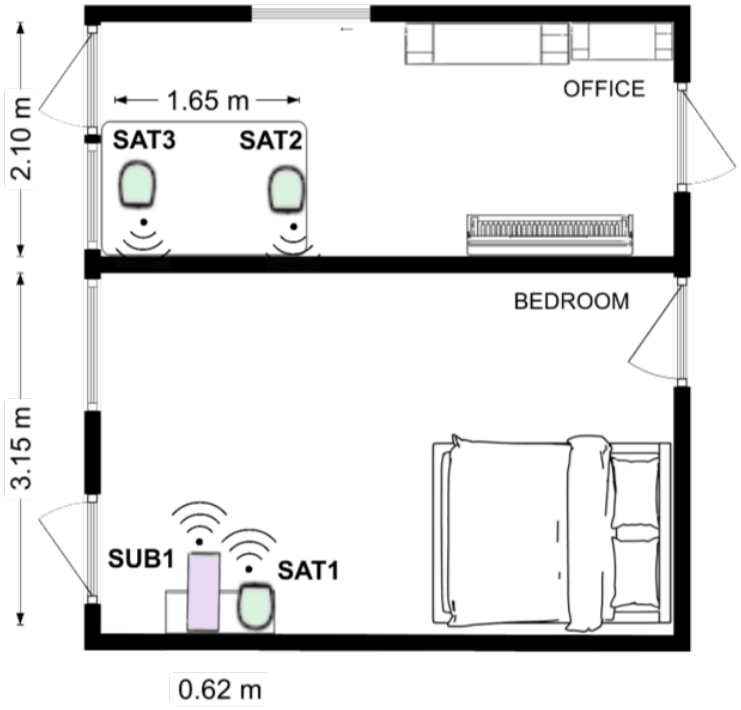


Figure D.4: Plaster walls environment

D.2.3 Bookshelf

Users will also place their speakers in and on top of certain furnitures, such as tables and bookshelf. This experiment simulates such a scenario. The setup is depicted in Figure D.6. For this experiment 20 % of the dataset was gathered for the class same room, as it shown in Table D.6.

Class label	Amount
Same room	5095
Different rooms	20489
Total	25584

Table D.6: Packet count ratio for all the classes in the environment bookshelf.

D.2.4 RF Disturbances

Every home is bound to experience some sort of RF frequency that will interfere with the systems. Some example are Wi-Fi devices (router and

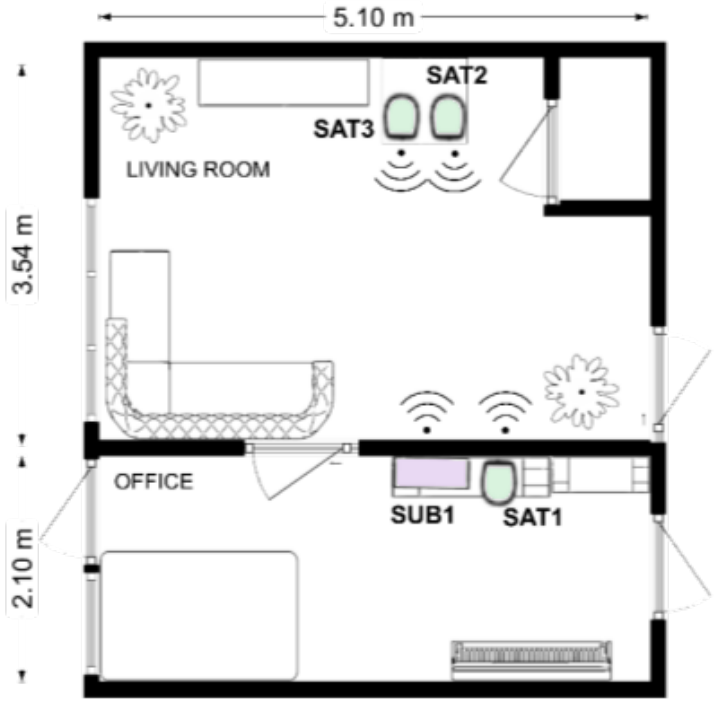


Figure D.5: Concrete walls environment

laptop), Bluetooth devices (smart phones), and other devices on the 2.4 GHz range (microwave). The experiment in Figure D.7 simulates such a scenario. Table D.7 highlights that only 34 % of the dataset is gathered for the same room class. This speakers in the same room had the most RF disturbances, and this could be a cause for the low packet reception.

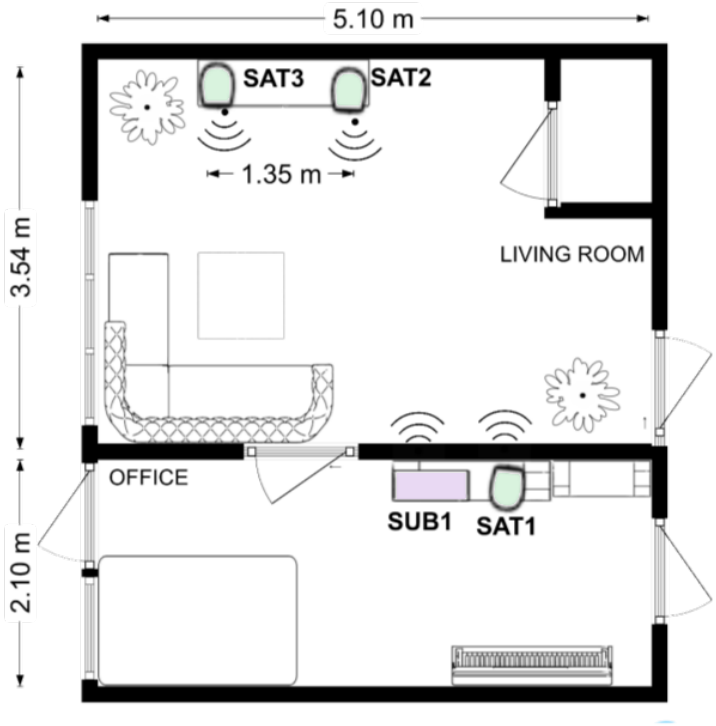


Figure D.6: Bookshelf environment

Table D.7: Packet count ratio for all the classes in the environment rf disturbance.

Class label	Amount
Same room	10967
Different rooms	21013
Total	31980

D.3 People

The purpose of this test was to see how people impacts the model. As people move around in a room this will cause extra variation. Additionally, people add attenuation to the RF signal.

D.3.1 Minimal Room

A baseline experiment was done in order to get an idea on how the RF behaves in a room with no people and with minimal furniture in a room. This is shown in Figure D.8. 35 % of the data samples was for the class *same room*, and the remaining 65 % were for the *different rooms*, as it is

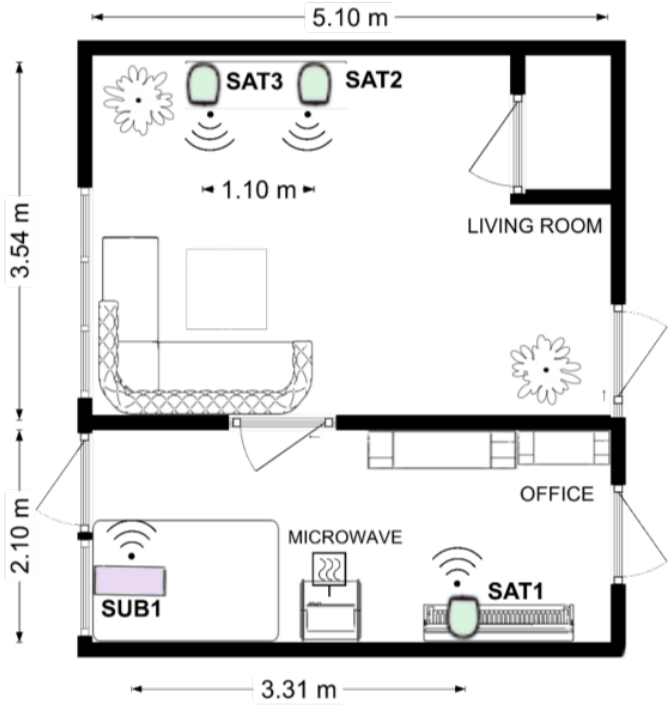


Figure D.7: RF disturbance environment

illustrated in Table D.8.

Table D.8: Packet count ratio for all the classes in the environment minimal room.

Class label	Amount
Same room	6724
Different rooms	12048
Total	18772

D.3.2 Person Walking

Experiments were done in order to be able to detect when a person is in a room. The goal was to see if by detecting a person in a room, the speakers in the same room of the user detect the person better than the speakers in different rooms. This experiment is depicted in Figure D.8. 33 % of the dataset were gathered for the class same room, as it is depicted in Table D.9.

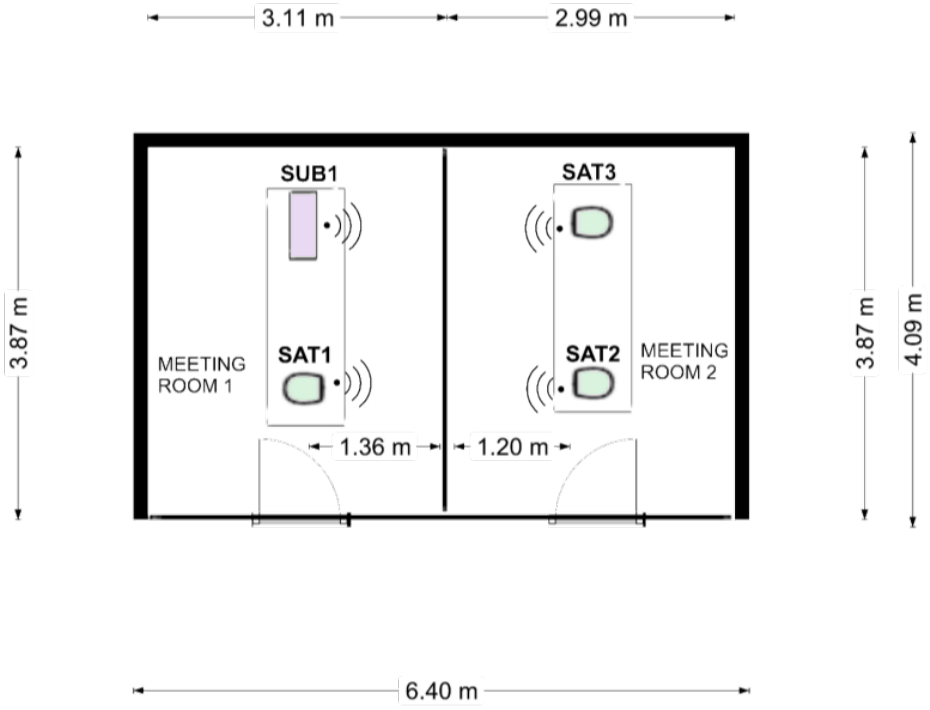


Figure D.8: Minimal room environment

Table D.9: Packet count ratio for all the classes in the environment person walking.

Class label	Amount
Same room	6579
Different rooms	13338
Total	19917

Appendix E

Shapiro-Wilk Test

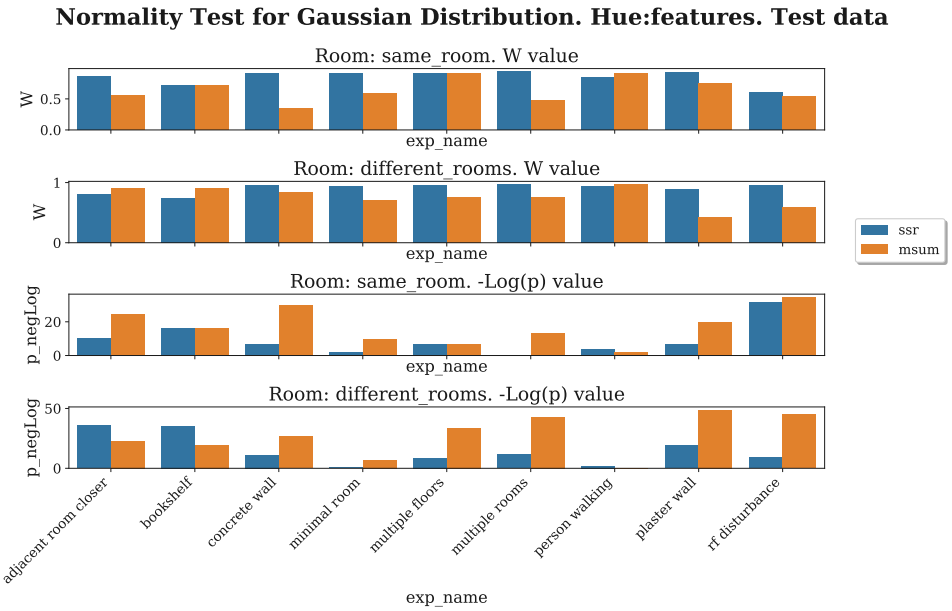


Figure E.1: Normality test for features' data distribution using the Shapiro-Wilk Test. For every environment the W and $-log_{10}(p)$ values are compared for all features per class label.

This page was deliberately left blank

Appendix F

Quantile-Quantile (QQ)-test

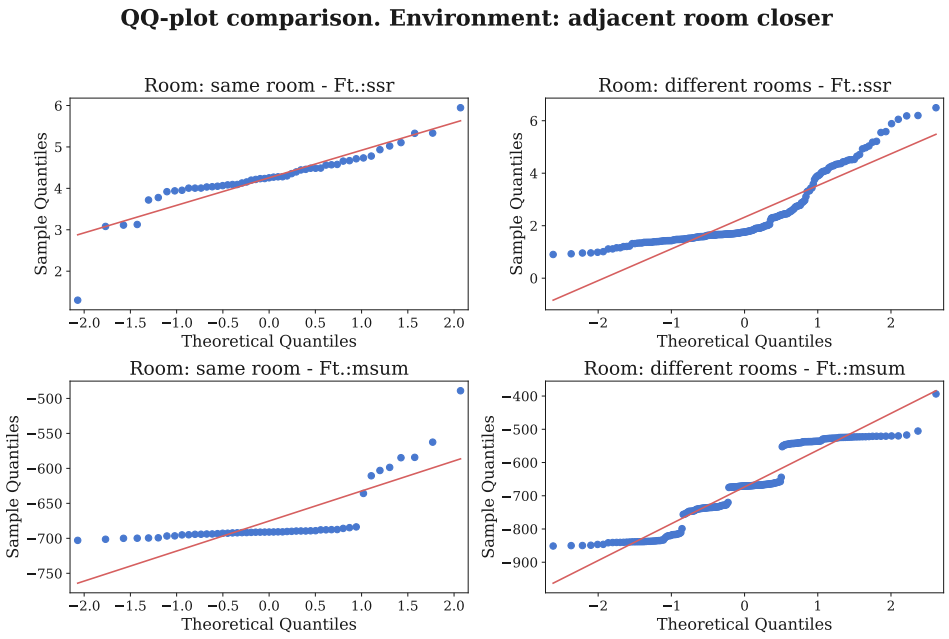


Figure F.1: QQ-test for the environment adjacent room environment.

QQ-plot comparison. Environment: bookshelf

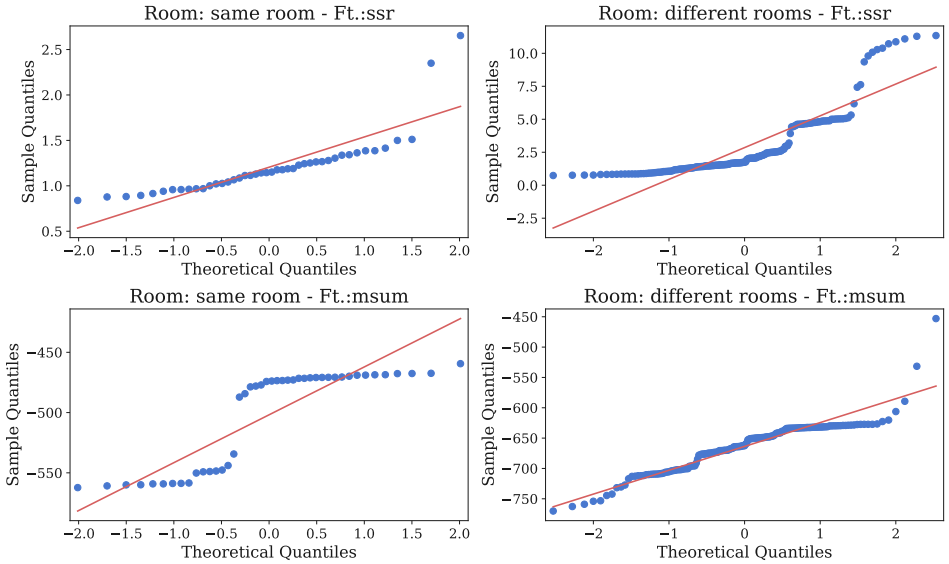


Figure F.2: QQ-test for the environment bookshelf.

QQ-plot comparison. Environment: concrete wall

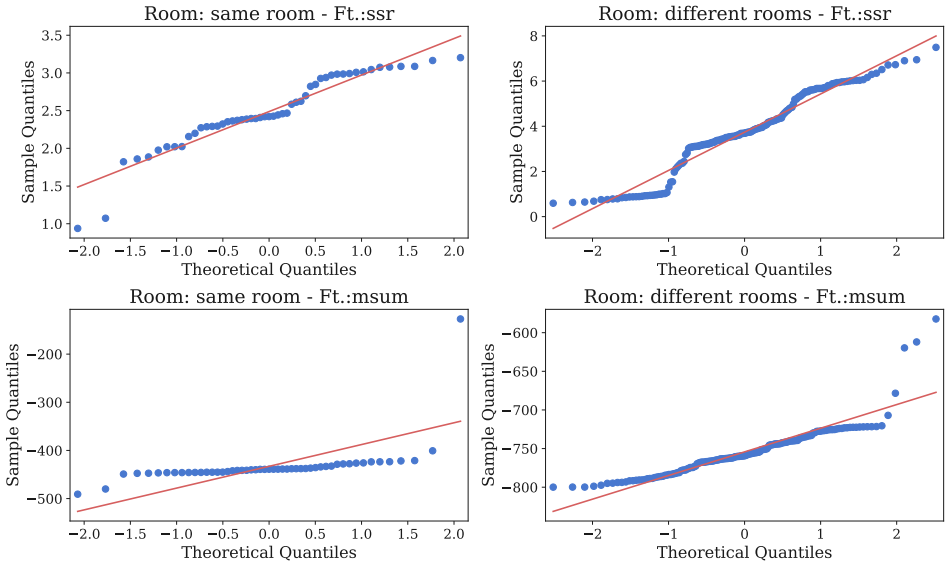


Figure F.3: QQ-test for the environment concrete wall.

QQ-plot comparison. Environment: minimal room

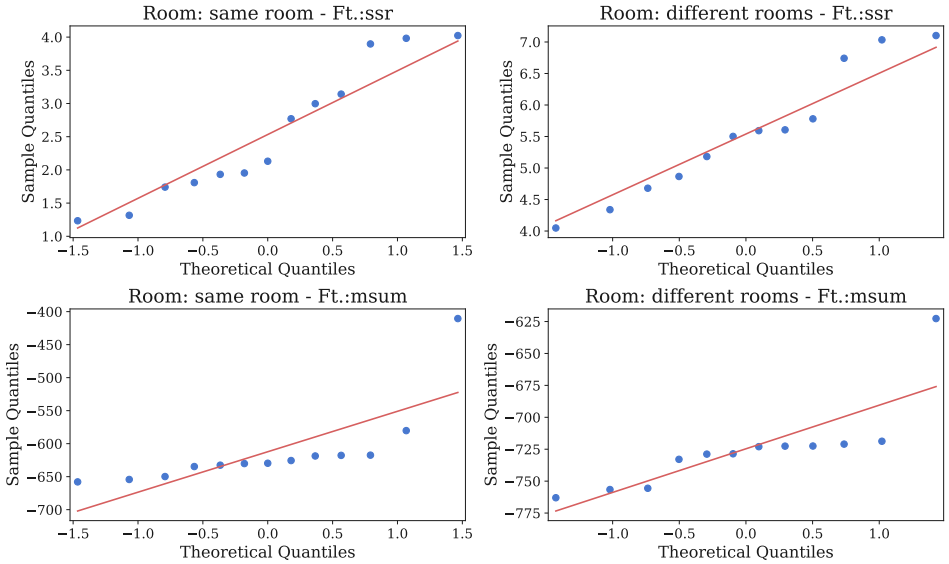


Figure F.4: QQ-test for the environment minimal room.

QQ-plot comparison. Environment: multiple floors

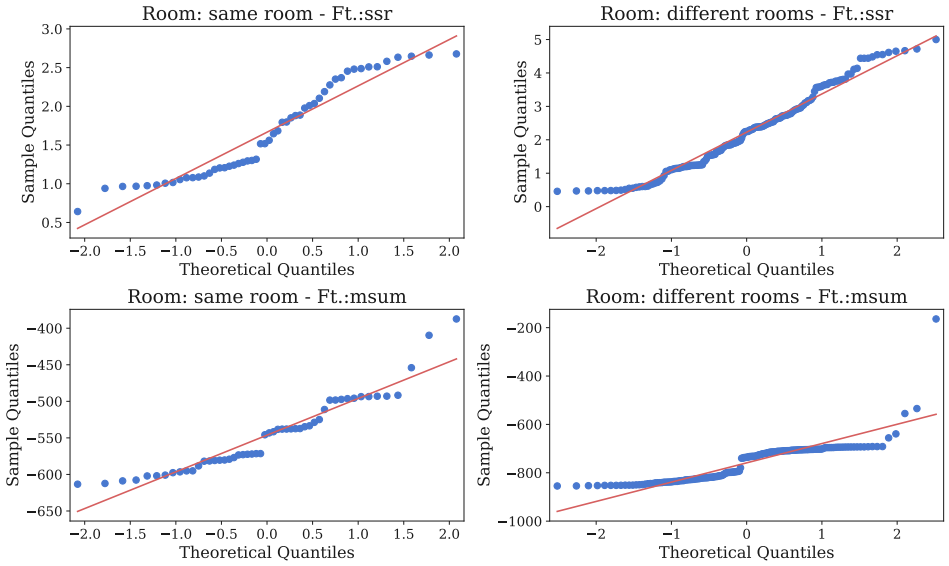


Figure F.5: QQ-test the environment multiple floors.

QQ-plot comparison. Environment: multiple rooms

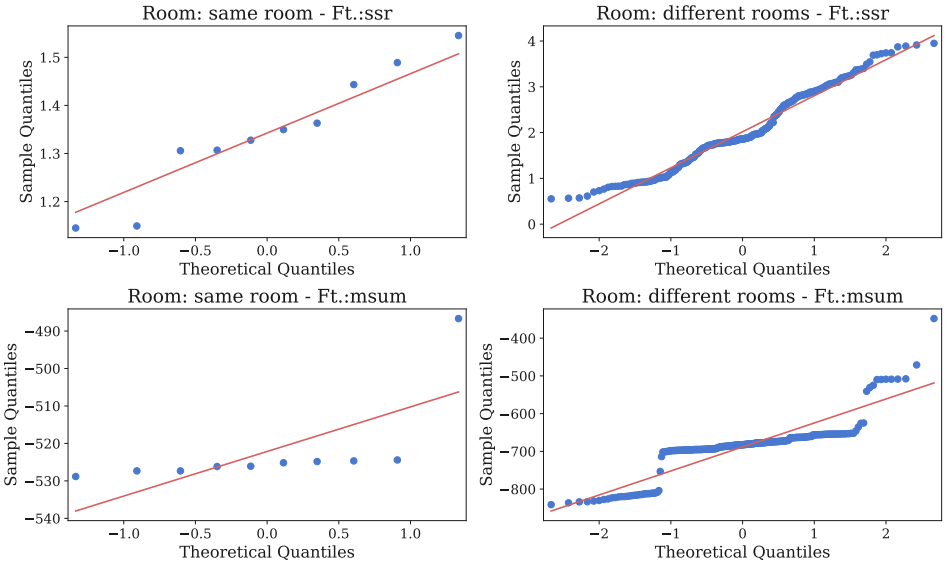


Figure F.6: QQ-test the environment multiple rooms.

QQ-plot comparison. Environment: person walking

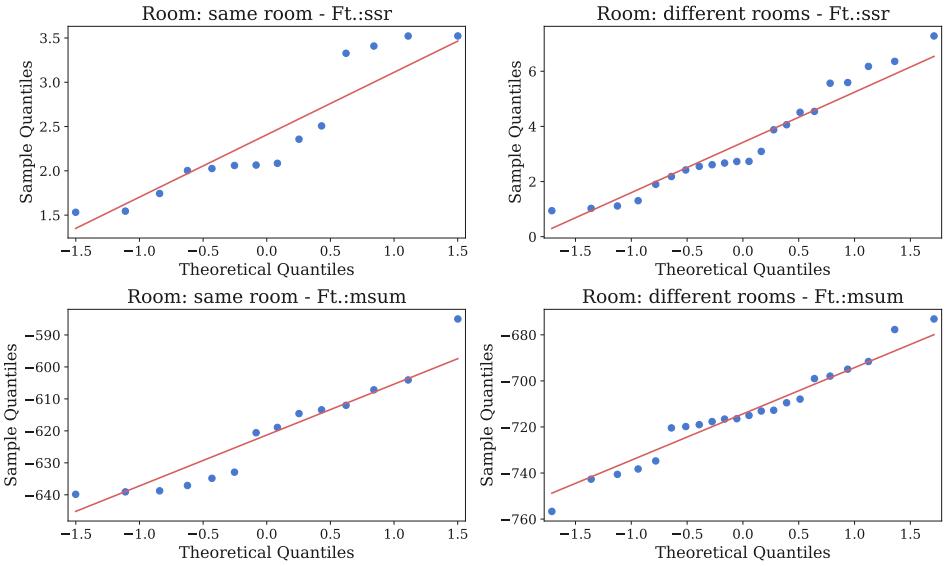


Figure F.7: QQ-test for the environment person walking.

QQ-plot comparison. Environment: plaster wall

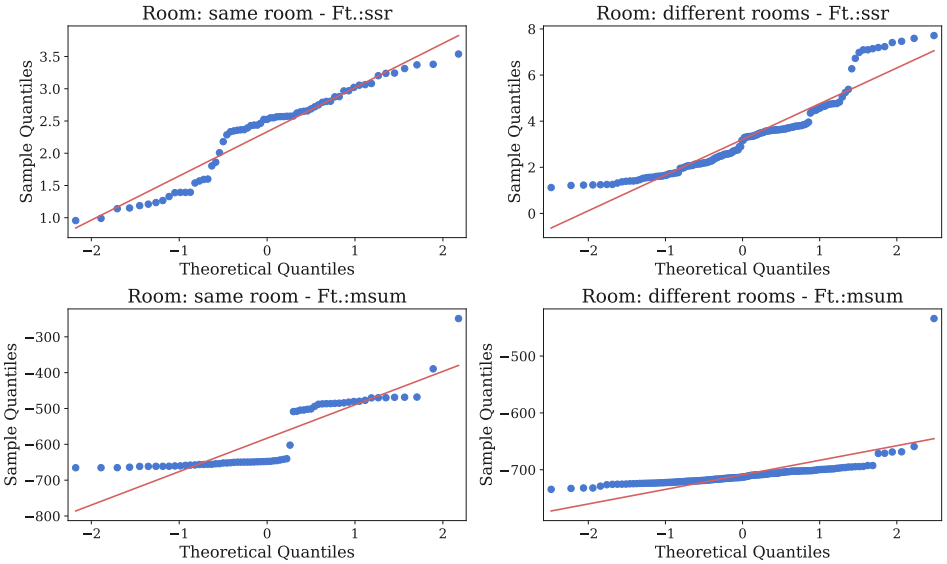


Figure F.8: QQ-test for the environment plaster wall.

QQ-plot comparison. Environment: rf disturbance

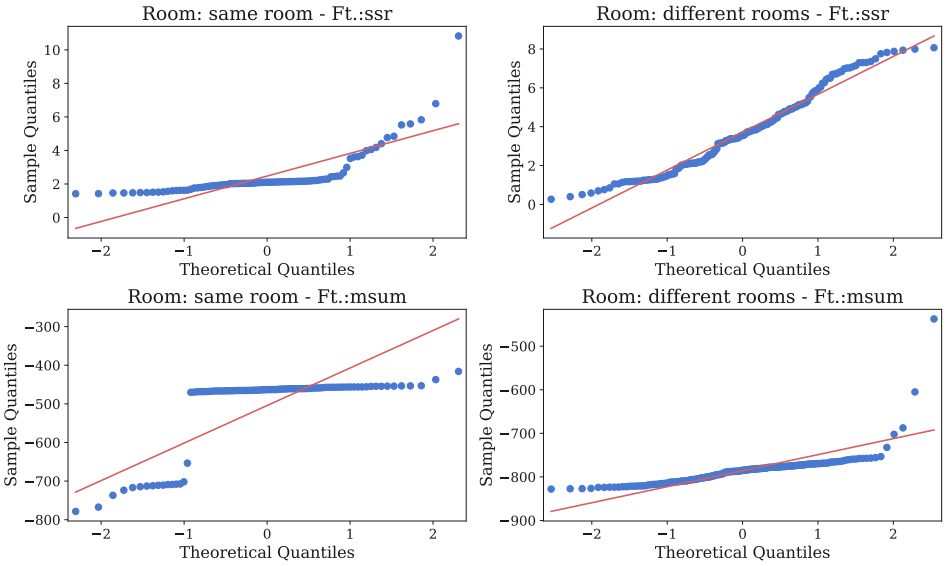


Figure F.9: QQ-test for the environment rf disturbance.

This page was deliberately left blank

Appendix G

Decision Boundary and Scatter Plot

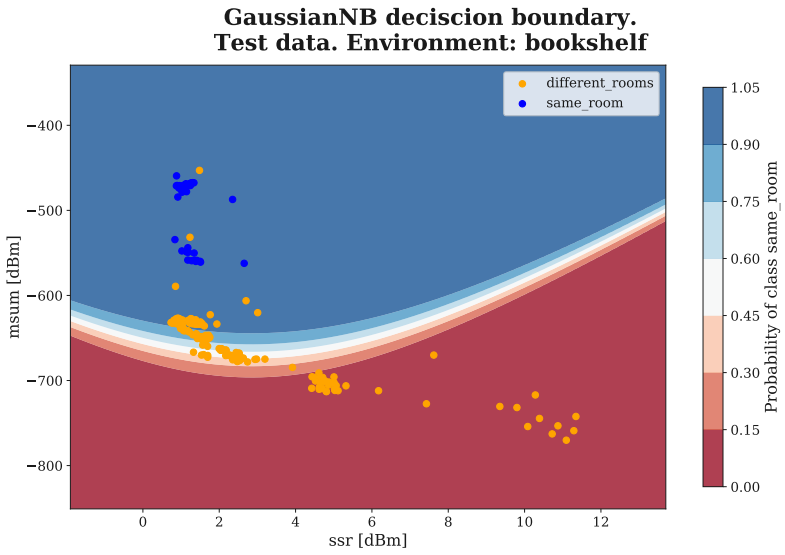


Figure G.1: NB Gaussian decision line and scatter plot for the environment bookshelf. A scatter plot is shown for each observation from the test data that belongs to either of the two classes, same room or different rooms. Every observation is given a probability of the likelihood of it belonging to the positive class, same room.

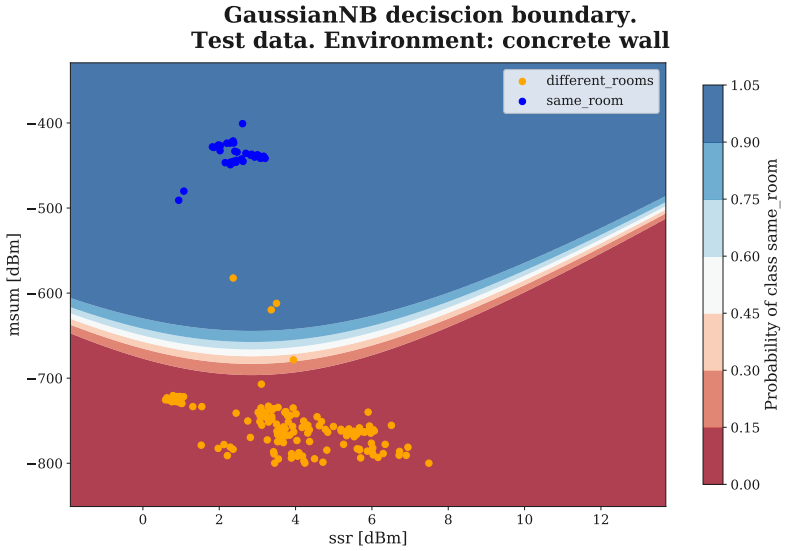


Figure G.2: NB Gaussian decision line and scatter plot for the environment concrete wall. A scatter plot is shown for each observation from the test data that belongs to either of the two classes, same room or different rooms. Every observation is given a probability of the likelihood of it belonging to the positive class, same room.

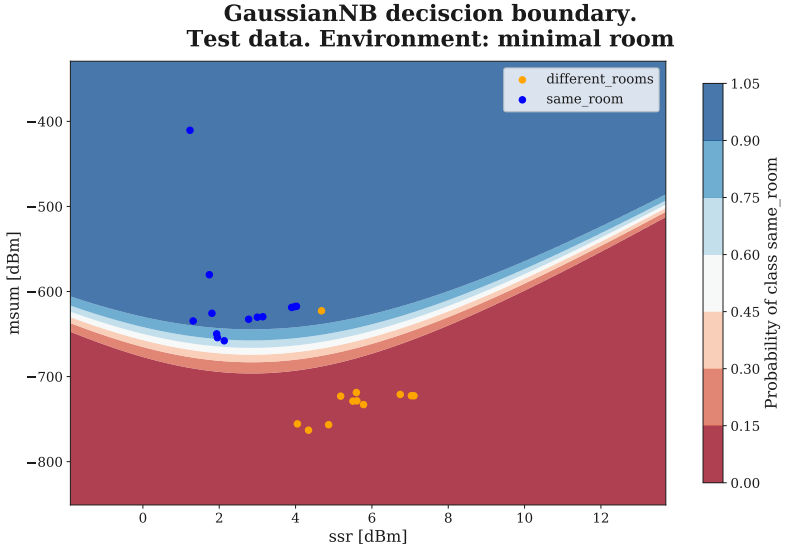


Figure G.3: NB Gaussian decision line and scatter plot for the environment minimal room. A scatter plot is shown for each observation from the test data that belongs to either of the two classes, same room or different rooms. Every observation is given a probability of the likelihood of it belonging to the positive class, same room.

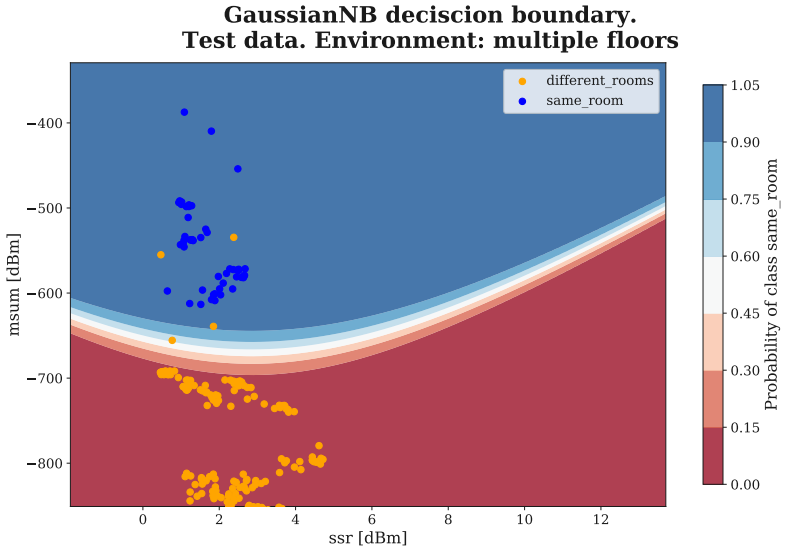


Figure G.4: NB Gaussian decision line and scatter plot for the environment multiple floors. A scatter plot is shown for each observation from the test data that belongs to either of the two classes, same room or different rooms. Every observation is given a probability of the likelihood of it belonging to the positive class, same room.

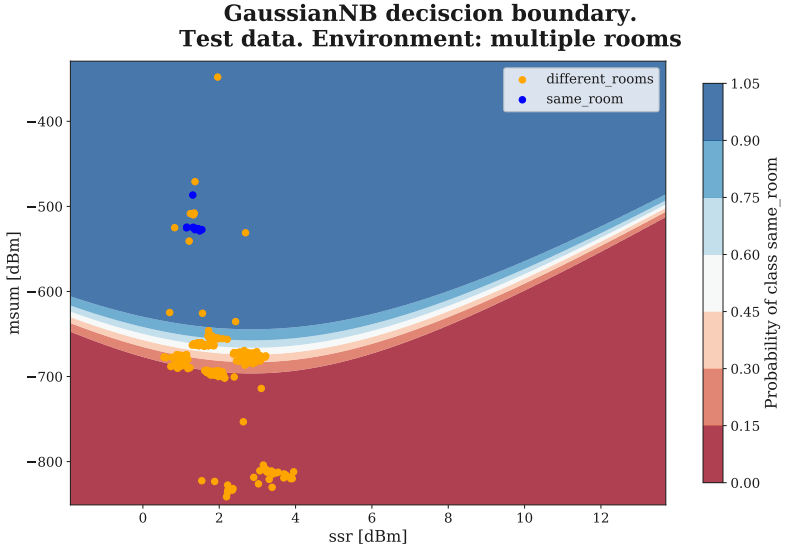


Figure G.5: NB Gaussian decision line and scatter plot for the environment multiple rooms. A scatter plot is shown for each observation from the test data that belongs to either of the two classes, same room or different rooms. Every observation is given a probability of the likelihood of it belonging to the positive class, same room.

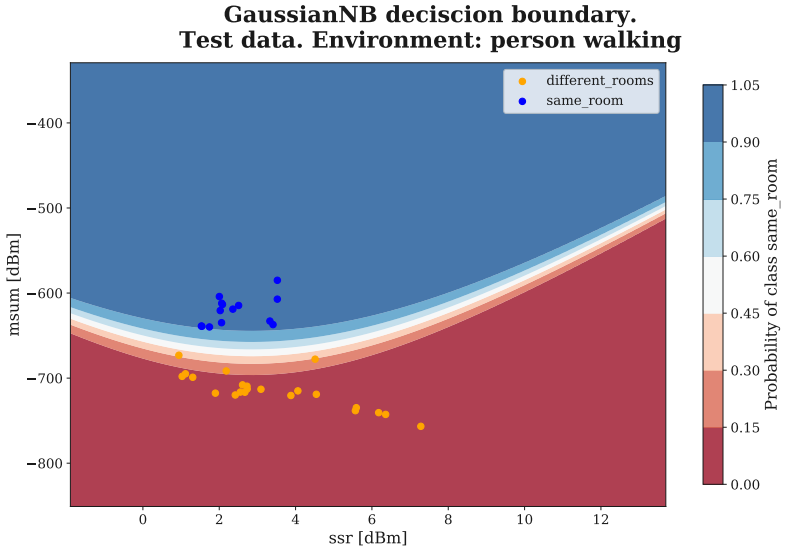


Figure G.6: NB Gaussian decision line and scatter plot for the environment person walking. A scatter plot is shown for each observation from the test data that belongs to either of the two classes, same room or different rooms. Every observation is given a probability of the likelihood of it belonging to the positive class, same room.

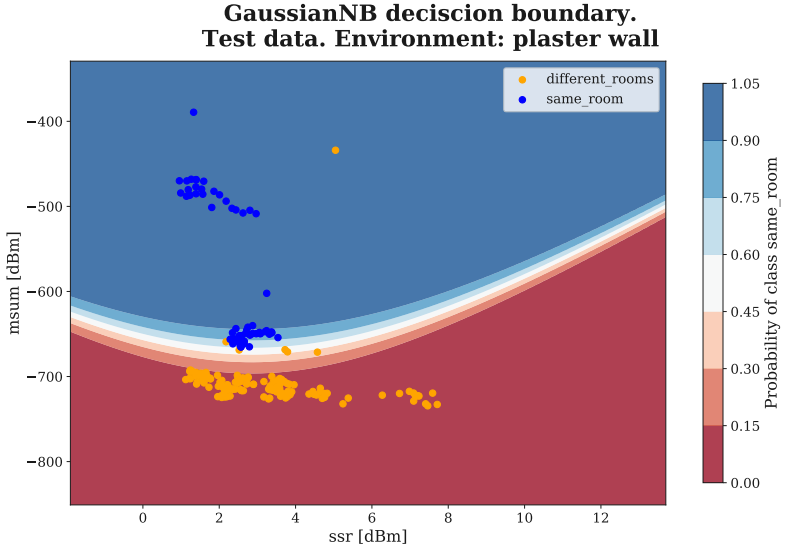


Figure G.7: NB Gaussian decision line and scatter plot for the environment plaster wall. A scatter plot is shown for each observation from the test data that belongs to either of the two classes, same room or different rooms. Every observation is given a probability of the likelihood of it belonging to the positive class, same room.

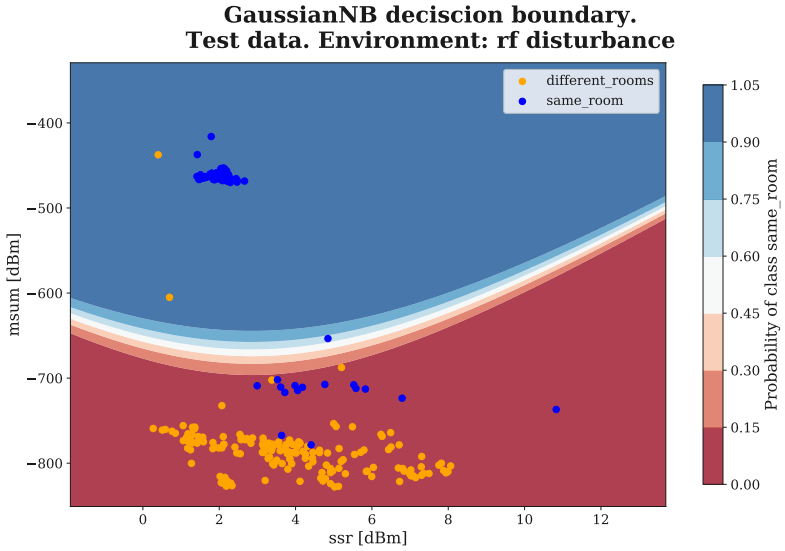


Figure G.8: NB Gaussian decision line and scatter plot for the environment rf disturbance. A scatter plot is shown for each observation from the test data that belongs to either of the two classes, same room or different rooms. Every observation is given a probability of the likelihood of it belonging to the positive class, same room.

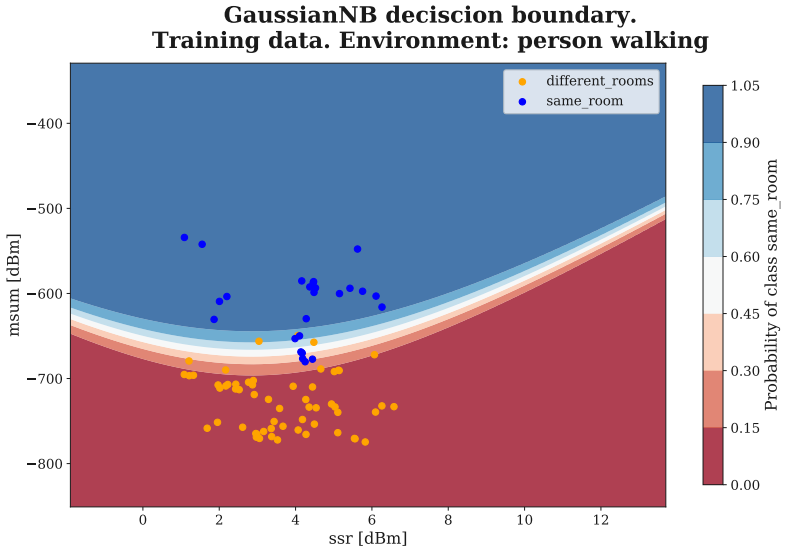


Figure G.9: NB Gaussian decision line and scatter plot for the environment person walking. A scatter plot is shown for each observation from the training data that belongs to either of the two classes, same room or different rooms. Every observation is given a probability of the likelihood of it belonging to the positive class, same room.

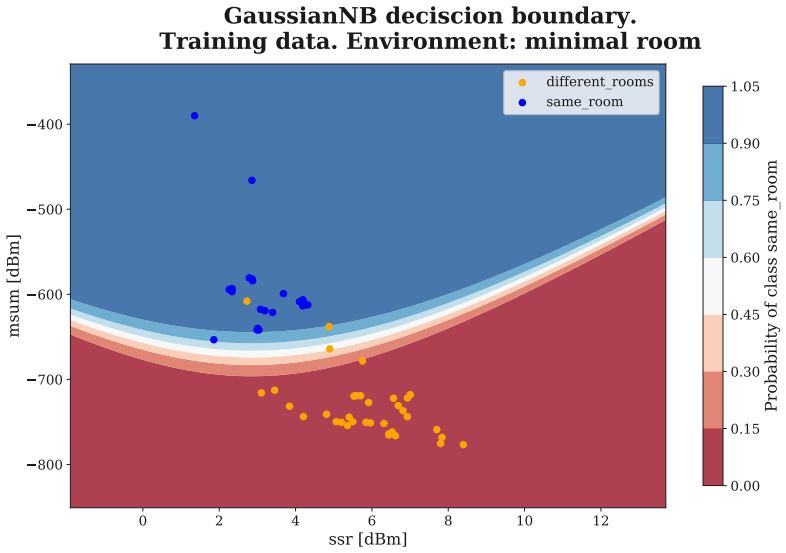


Figure G.10: NB Gaussian decision line and scatter plot for the environment minimal room. A scatter plot is shown for each observation from the training data that belongs to either of the two classes, same room or different rooms. Every observation is given a probability of the likelihood of it belonging to the positive class, same room.

Appendix H

Features' Data Distribution

H.1 Box plot

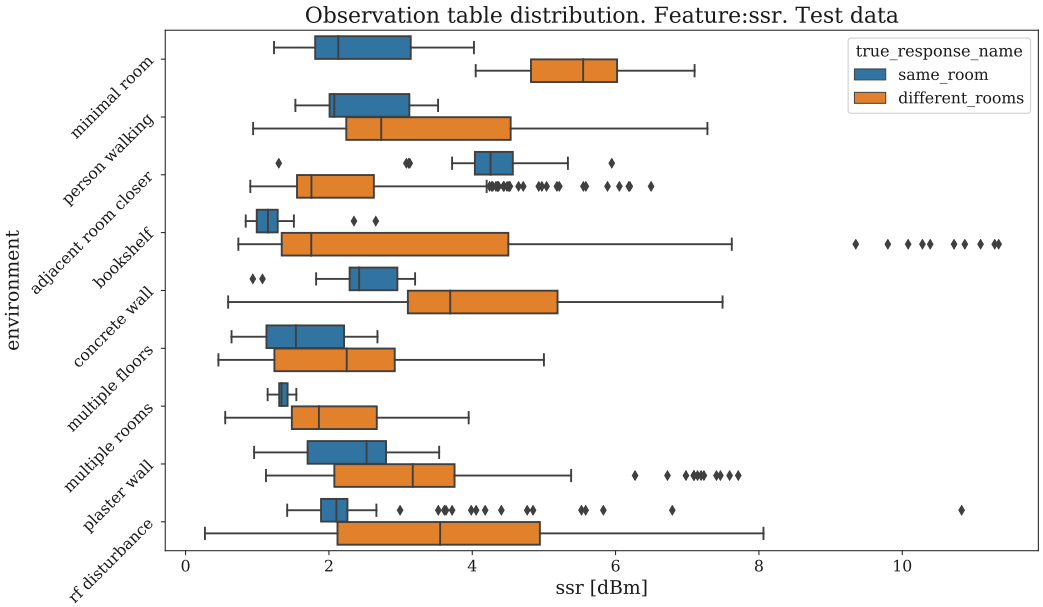


Figure H.1: Box plot describing the distribution of the feature ssr using the test data. For every environment the distribution is compared for both classes, same room versus different rooms.

.