Norwegian University
of Life Sciences

Master's Thesis 2020 30 ECTS
Faculty of Science and Technology (RealTek)

Sampling strategies to estimate Noise level with regards to the Energy-Accuracy Trade-Off

Emil Skar
Industrial economics

Preface

Thank you. This is the recurring theme in my preface. Thank you. Thank you to my brother Josef for making me laugh and for being a viking and helping me move when the thesis' deadline was looming. Thank you to my roomies, Alan & Vignaa & Ulrikke & Thach, for keeping up with my mood changes, loving me unconditionally and being captains when i needed it. I want to thank my mother, for taking care of my mental state and always providing me with positive vibes.

Thank you to my supervisors, Associate Professors Oliver Tomic and Kristian Hovde Liland at the Faculty of Science and Technology (REALTEK). I want to thank you on behalf of all your students for all the work you put in, both seen and unseen. You supported me the whole way and I'm very thankful for the Zoom-meetings which made me keep my sanity.

Thank you to Jon Nordby and the Soundsensing team for all your help. Jon, your guidance has been invaluable, and I'm in awe of your knowledge. Seriously. Thank you to Faiga Alawad and Frank Kramer for your support and for sharing your data unselfishly with me. Thank you to Fabian Nemazi for peer reviewing parts of my code. Thank you to all my day ones from Haugenstua.

And lastly, yet very importantly,

Thach til Slangehulen

Ås, 30th June, 2020

Emil Skar

Abstract

Noise is a growing problem in today's society. This is especially true in urban areas, where noise pollution has become an important factor in the deteriorating health of the residents. Due to this, several regulations have been implemented by different governmental bodies. One of these is EU's directive 2002/49/ec, which says that noise maps must be created for specific areas that are especially noise heavy. These noise maps are created through simulations and have been shown to be imprecise and also dependent on good data sources, something which is scarce. Innovations look at noise monitoring through wireless communication systems to counteract these problems, but they are mostly dependent on either a cable grid for energy supply or a secondary energy source.

Most of the noise indicators used today are cumulative of nature, and thus a continuous data stream is not of necessity. Therefore, a thought is to introduce a sampling strategy to the wireless sensors. A sampling strategy decides when to measure sound and when not to, and may lessen energy usage.

A dataset containing continuous measurements over 11 weeks of a student working environment at NTNU is used to evaluate different sampling strategies. The dataset consists of data collected from five sensors. The data was preprocessed and a master sensor chosen. The rest of the sensors' data was utilized to impute the missing values of the master sensor. The time interval of noise measurements was chosen to be 15 minutes.

Three algorithms; a Dummy Regression, a Linear Regression and a Random Forest Regression were trained and evaluated. The target variable was the RMSE between the $L_{Aeq,15min}$ containing all measurements and the $L_{Aeq,15min}$ containing only the subsampled measurements.

The models' performance was put in the context of the economic benefits that a lower energy usage may give. A Pareto-front was used to find the optimum, and it was concluded that a subsampling rate of 65 % was optimal for the student working space. A cost-benefit analysis was done on four different sensor network alternatives consisting of 22 sensors, and the best scoring alternative was the one that implemented the subsampling rate that NTNU SOA recommended.

Sammendrag

Støy er et økende problem i dagens samfunn. Dette gjelder spesielt for urbane områder, der støyforurensning har blitt en viktig faktor i den svekkede helsen til innbyggerne. På grunn av dette er flere forskrifter blitt lagt frem av forskjellige statlige organer. Et av disse er EUs direktiv 2002/49 / ec, som sier at det må lages støykart for spesifikke områder som er spesielt støytunge. Disse støykartene er laget gjennom simuleringer og har vist seg å være upresise og også avhengige av gode datakilder, noe som er mangelvare i dag. Innovasjoner ser på støyovervåking gjennom trådløse kommunikasjonssystemer for å motvirke disse problemene, men de er stort sett avhengig av enten et kabelnett for energiforsyning eller en sekundær energikilde.

De fleste støyindikatorene som brukes i dag er kumulative av natur, og derfor er det ikke nødvendig med en kontinuerlig datastrøm. Derfor er en tanke å introdusere en samplingsstrategi for de trådløse sensorene. En samplingsstrategi bestemmer når du skal måle lyd og når ikke, og kan redusere energiforbruket.

Et datasett som inneholder kontinuerlige målinger over 11 uker i en gruppearbeidsplass ved NTNU brukes til å evaluere forskjellige samplingstrategier. Datasettet består av data samlet inn fra fem sensorer. Dataene ble preprosessert og en hovedsensor valgt. Resten av sensorenes data ble brukt til å imputere de manglende verdiene til hovedsensoren. Tidsintervallet for støymålinger ble valgt til å være 15 minutter.

Tre algoritmer; en dummy-regresjon, en lineær regresjon og en random forest regresjon ble trent og evaluert. Målvariabelen var RMSE mellom $L_{Aeq,15min}$ som inneholder alle målinger og $L_{Aeq,15min}$ som bare inneholdt de samplede målingene.

Modellenes ytelse ble satt i sammenheng med de økonomiske fordelene som en lavere energibruk kan gi. En Pareto-front ble brukt for å finne optimum, og det ble konkludert med at en samplingrate på 65 % var optimal for studentens arbeidsplass. En kostnad-nytte-analyse ble gjort på fire forskjellige sensornettverksalternativer bestående av 22 sensorer, og det beste scoringsalternativet var det som implementerte sampling strategien som NTNU SOA anbefalte.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Noise

”The only constant in life is change” said Heraclitus, a Greek philosopher who lived around 500 BC. The author of this master thesis would argue that another constant in life is noise. This has become especially true in the last decades, where the urbanization of the world and human proclivity for surrounding themselves with ceaseless stimulation have made noise not only a constant but also a problem. In western Europe alone, it is estimated that a minimum of one million deaths come from traffic-related noise [1]. This warrants the question: ”What is noise?”. When researching noise it is essential to define the difference between noise and sound. Noise is a subset of sound which is unwanted. The unwantedness of the specific sound depend on the subject which listens, so it is important to properly define what is sound and what is noise.

To combat the growing problem of environmental noise EU, the European Union put a directive forward in 2002. The directive made it mandatory for EU member countries to create noise maps for all railways, airports, major roads and urban areas over a specific size [2]. A noise map is a map that shows the sound levels and distribution over a given area. The noise maps only capture yearly averages and have to be updated every five years, as per the directive. The noise maps are made through simulations determined by predetermined parameters such as traffic flow and 3D-terrain. Mioduszewski et al. [3] have shown that these simulations to not adequately describe the real soundscape in the areas. The simulations are dependent on stable data sources and the associated data collection methods, which is not always available. The data sources may be sensors placed in the street network, models of the street network and digital terrain models. Furthermore, recent research has shown that noise levels consistently exceed the limits recommended by the EU [4]. The demands of the regulative bodies combined with the challenges

today's methods experience in precisely describing the soundscape has led to the development of alternative methods. A crucial part of this includes real-time monitoring using wireless sensors to estimate sound levels.

1.2 Noise monitoring with wireless systems

Due to the expensive equipment and the inflexibility of today's systems, there is a huge incentive in innovating the noise monitoring industry. Several projects use wireless sensors to more efficiently estimate sound levels and classify sound sources. Some examples of these are SONYC [5], SENSEable [6], DYNAMAP [7] and CENSE [8]. A Norwegian startup that has received national attention is Soundsensing [9]. Soundsensing uses a sensor network consisting of self-developed sensors with embedded machine learning models done on-edge (in the wireless sensor package), to minimize the information sent over the IoT-network. On-edge means that the machine learning models are run locally on the sensors, and preferably only the result is transmitted over the IoT-network.

1.3 The economical costs of noise

The economical cost of noise is not only about the aforementioned deaths and illnesses, which may be induced by noise. There is also the question about the viability of sensor companies that focus on noise mapping.

With continuous real-time monitoring of sound levels, the IoT-device has to be constantly in use. For noise estimation, many metrics such as L_{den} and L_{night}, which are defined in 2.1.5, are only reported as an average per day, and it may have the potential to skip/reduce the data acquisition. For humans, a change in noise of ± 3 dB is the lowest that they can sense. ± 10 dB is observed as a doubling or halving of sound loudness, while ± 5 dB is deemed as audibly recognizable [10, 11]. This leads to the decision to use ± 3 dB as a boundary for acceptable error, as errors under this threshold are not considered as noticeable by humans.

This leads to the question, "How often does the IoT-device need to record noise levels while keeping the accuracy below the error threshold?" The trade-off between the model's accuracy in predicting the noise level error and battery usage will be key to answer this question, and a Pareto-front will be used to find the optimum point. There is already some existing work on the topic of noise estimation and sampling strategies. Two essential resources in this realm are [12] and [13], which utilize the same dataset as in this master thesis.

The noise estimation analysis is done on indoor data in Koopen, but a cost-benefit-analysis is done on an imaginary installation with 22 sensors in a Wireless Sensor Network in an outdoor environment. This because the dataset of interest is on an indoor study environment, but the big motivation is to try to extrapolate the findings

done on the indoor dataset to an outside space. A reduction in energy consumption can create business cases that previously were not economically viable due to the economic cost. With this in mind, municipalities and national governing bodies may take a more proactive stance on noise and implement solutions based on real-time data.

1.4 Problem scope

The problem scope is limited to using different sampling strategies on a master sensor in a student working space and predicting the error of the noise level. The scope will also be limited to only the sampling strategy. Furthermore, the results of the predictions will be applied in an economic analysis where a Pareto optimum and Cost-benefit analysis are applied.

1.5 Research questions

1. How often does the IoT-device need to record noise levels, while keeping the accuracy below the error threshold with regards to the energy-accuracy trade-off?
2. In a Wireless Sensor Network consisting of 22 sensors, which alternative with the associated sampling strategy and energy usage is the best with regards to net value?

1.6 Structure of the thesis

This thesis starts with the theory behind sound, machine learning, microcontrollers and the performed economic analysis in chapter 2. Chapter 3 discusses the current state of arts and sampling strategies. Chapter 4 explores and preprocesses the dataset, and the methodology used is described. Chapter 5 presents the results which is further discussed in chapter 6. Lastly, the findings in this thesis is summarized in chapter 7.

Chapter 2

Theory

This chapter presents the theory regarding the topics relevant to this master thesis. The topics covered will be sound, machine learning, microcontrollers and economic analysis.

2.1 Sound theory

Sound can be described as variations in pressure over time [14]. To measure sound, both the amplitude of the sound pressure wave as well as the frequency of the waves are recorded. An example of a sound wave and the two mentioned metrics is illustrated in Figure 2.1 on the following page. The amplitude decides the loudness of the sound. The greater the amplitude of the sound, the greater the energy of the sound, and the louder it is perceived. The frequency affects the pitch of the sound. The pitch of a sound can be described as how the ear and, subsequently, the brain, interprets the sound. A sound with high frequency will have a high pitch, while a sound with a low frequency will have a low pitch. Humans generally can hear in the frequency range of 20 Hz to 20 kHz. The human ear is the most sensitive in the frequency range from 2 kHz to 5kHz [15].

The intensity of a sound wave is the amount of energy that is transported over an area. When the amplitude is bigger, more energy is transported, and the intensity of the sound wave is larger. The intensity can be expressed through $Intensity = \frac{Energy}{Time * Area}$ or $Intensity = \frac{Power}{Area}$ since $\frac{Energy}{Time} = Power$. A common unit for expressing a sound wave's intensity is $Watts/meter^2$. The area the sound wave it is transported over and its intensity is an inverse square relationship [17]. This means that when the distance from a sound source has doubled, the intensity is reduced to one fourth.

When researching noise, it is important to define the difference between noise and

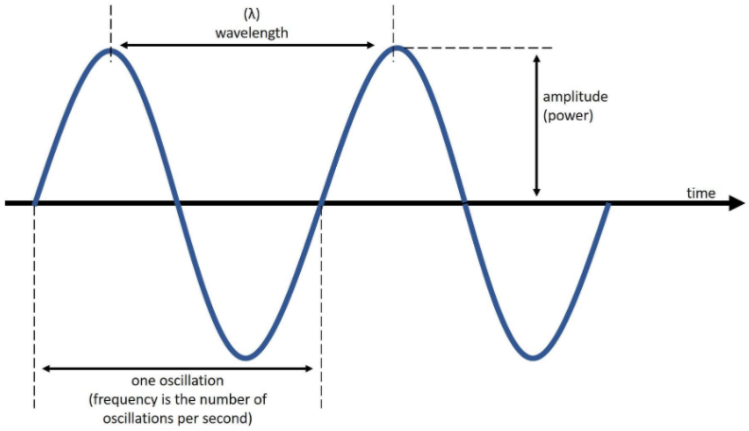


Figure 2.1: *Image of a sound wave showing the amplitude and frequency of the pressure wave over time. Adapted from [16].*

sound. Noise is a subset of sound which is unwanted. The unwantedness of the specific sound may be dependent on the subject (human) which listens, so it is crucial to properly define what is sound and what is noise. Noise has been shown to affect students negatively when studying and affect the communication, intelligence and focus of the student [18].

2.1.1 Measuring loudness

The loudness of a sound is typically measured in decibel. Decibel is a logarithmic dimensionless unit described by the ratio between the measured and the defined reference quantity for the unit. Decibel may be used to describe power levels, currents or in this master thesis' case: sound pressure levels.

The formula used for decibel is:

$$L = 10\ log_{10}(\frac{A}{B})\ dB \tag{2.1}$$

where L is expressed in decibel and represents the sound level, A is the measured quantity, while B is the reference quantity. The measured quantity may be the measured Sound Pressure Level, while the reference quantity may be Sound Pressure Level of the threshold of hearing. In this thesis, the L represents the Sound Pressure Level (SPL).

2.1.2 Threshold for error

The lowest sound a human ear can perceive is set to 0 dB. 0 dB is equivalent to an intensity of $1 * 10^{-12} W/m^2$. A sound that is ten times more intense is given a dB value of 10. Due to the nature of the logarithmic scale, a 10 dB increase of sound will be experienced as a two-fold increase in sound volume. While the sound volume is experienced as a doubling in volume, the sound energy is experienced as a ten-fold increase. When an object vibrates, it produces a kind of mechanical energy. This is sound energy. A doubling in sound energy will be experienced as an increase of 3 dB by a human's ears. If a human is exposed to a continuous sound for 1 hour at a volume of 40 dB, the same person would only need to stay in the room for 30 minutes to be exposed to the same amount of energy if the volume was 43 dB. Noise limits are often based on this, and in the example this is a 3 dB exchange rate [14]. To be exposed to more sound energy is a larger load on the ears even if the comparative decibel difference is not that great. The effect this has on a human is also influenced by the aforementioned distance, and thus intensity, between the sound source and the person exposed to the sound.

This leads us to a definition of a threshold for error for the predictions going to be made later in this thesis. ± 10 dB is observed as a doubling or halving of sound loudness, while ± 5 dB is deemed as audibly recognizable. A change in the noise of ± 3 dB is the lowest that humans can sense [10, 11]. A decision to use ± 3 dB as an boundary for acceptable error is made, as errors under this threshold are not considered to be noticeable by humans.

2.1.3 Frequency Weighting

To mimic the human hearing system, one can apply weighting filters. Some of these are standardized in IEC 61672-1:2013 [19].

A-weighting is one of these filters and is commonly used when looking at industrial and environmental noise. Without any loss of hearing abilities humans can hear in a range from 20 Hz to 20 kHz, and are most sensitive in the range from 2 kHz to 5kHz. These are the frequencies that A-weighting emphasises. C-weighting is another type of frequency-weighting. In C-weighting, the low-frequency sounds are more emphasized than in A-weighting, and it measures in the frequency range of 30 Hz to 10 kHz, while A-weighting measures in the frequency range of 500 Hz to 10 kHz. The peaks of noise are emphasized in C-weighting. A third method is the Zwicker method [20]. This method measures the binaural loudness. The threshold for a sound heard by two ears is usually lower than just for one ear.

In this study, A-weighting is used. To show that the sound has been A-weighted, an A is added to the dB suffix, dBA. It may also be shown as L_{Aeq}.

2.1.4 Overview of different sound levels

Different type of noise has different dBA levels. To build an intuition of the different thresholds of dBA, which the human ear detects, Table 2.1 is presented. A quiet library is estimated to have a dBA of 40, while a noisy restaurant may have a dBA of up to 85. Hearing loss may occur when exposed to a dBA of over 85 over a more extended period.

Table 2.1: *Overview of different sound sources and their accompanying dBA level. Distance from sound source where applicable. Adapted from [12]*

Sound source	dBA
Human hearing threshold	0
Breathing normally (1m)	10
Whispering (1m)	30
Quiet library	40
Large office, busy street (90m)	60
Normal conversation (1m)	65
Vacuum cleaner (3m)	70
Heavy traffic, noisy restaurant	85
Truck (10m), shouted conversation (1m)	90
Chainsaw (1m)	110
Rock concert (5m), threshold of discomfort	120
Jet engine (50m)	130
Threshold of pain	140
Gunshot (0.5m)	160
Explosion (close)	190

2.1.5 Indicators of noise

The aforementioned EU Directive [2] operates with two main noise indicators, L_{den} and L_{night}. These two are based on L_{Aeq}, and most research studies use L_{den} and L_{night} to determine noise levels. L_{den} is the average noise level for a day. The acronym den stands for day-evening-night. The day is split into three, where the day-period is from 07-19, evening-period from 19-23 and night-period from 23-07. Added together $Lday + Levening + Lnight = Lden$. $Lday$ is a day-weighted noise-indicator, $Levening$ is an evening-weighted noise-indicator while L_{night} is a night-weighted noise-indicator. The evening and night periods have a weighting,

respectively 5 dBA and 10 dBA, which are added to the period's dbA-value. This is due to human's increased sensitivity for noise in these time frames. To get the values of e.g. $Lday$ is not possible to just do a normal average of the decibel values due to the decibel formula's logarithmic nature. Thus, to calculate $L_{Aeq,T}$, where T is a given time-period the formula is:

$$L_{Aeq,T} = 10\ log_{10}\left[\frac{\frac{1}{T}\int_{t_1}^{t_2} p_A^2(t)dt}{p_0^2}\right]\ dB \tag{2.2}$$

where $pA(t)$ is the instantaneous A-weighted sound pressure at running time t, and $p0$ is equal to 20 µPa (micropascals).

In the dataset used in this master thesis L_{Aeq} is already provided, not only sound pressure levels, so instead another formula has to be used. The anti-log of the values are added together and divided by the number of samples *n*.

$$L_{Aeq,T} = 10\ log_{10}\left[\frac{1}{n}\sum_{i=1}^{n} 10^{\left(\frac{L_{Aeq,1s}}{10}\right)}\right]\ dB \tag{2.3}$$

To capture a different representation of the noise, one can look at the statistical distribution of $L_{Aeq,T}$ for the time interval T. Some possible indicators are $L_{A10,T}$, $L_{A50,T}$ and $L_{A90,T}$. $L_{A10,T}$ looks at the peaks of noise and is the A-weighted SPL exceeded for 10 % of the time interval T. This noise indicator is often used for noise measurement of traffic. $L_{A50,T}$ describes the average/median of the noise and is the A-weighted SPL exceeded for 50 % of the time interval T. $L_{A90,T}$ describes the background noise level and is the A-weighted SPL exceeded for 90 % of the time interval T.

2.1.6 Time interval of noise measurement

Noise is such that it often is the sustained noise levels that affect the health and hearing of humans. Therefore an integral part of an experiment looking at noise pollution is the time interval in which to accumulate the noise measurements. Research by Brocolini et al. [21] has shown that a 10 minute time interval of noise measurements gives a representative showing of the noisescape in almost all cases. If there is more variability in the noise measurements, longer time intervals may be needed. This is supported by the the Norwegian Environmental Agency (Miljødirektoratet) guideline on noise measuring for industries. A 10 minute interval was proposed here as well, and in the case of large variability in noise measurements, the noise measuring should be repeated hourly [22]. The repetitions are set to a minimum of three in the nighttime and five in the daytime. In [12] different time intervals were tested and how the sampling affected the associated noise indicators. A more robust noise indicator is less affected by the sampling

strategy implemented. The ranking of the robustness of the noise indicators was as following:

1. $L_{Aeq,8h}$
2. $L_{A50,15min}$
3. $L_{A90,15min}$
4. $L_{Aeq,1h}$
5. $L_{Aeq,15min}$
6. $L_{A10,15min}$
7. $L_{A10-A90,15min}$
8. $L_{NPL,15min}$

In Kraemer et al. [13] $L_{Aeq,15min}$ was used as the noise indicator.

2.2 Preprocessing and machine learning

To pick the optimal sub-sampling percentage, machine learning will be used. Machine learning is the science in which algorithms are fed data and train models that perform a task without explicit programming. To grasp what kind of tasks can be solved with machine learning, two simple examples are provided. One is credit card transactions and the other postal code readings. When a credit card transaction is done through a website, it is vital for the company that it is not fraudulent. Here, the machine learning task would be to determine if the transaction is likely to be fraudulent based on the information collected about the user up to the point of the transaction. In the scenario of postal code readings, the machine learning task may be even more straight-forward and be to determine what the postal code on a letter is based on a snapshot of the letter. Both these tasks can be and are solved with machine learning today.

Machine learning has different sub-genres. These are normally split into three: supervised learning, unsupervised learning and reinforcement learning. The difference between these sub-genres lies in how the algorithms learn.

In supervised learning, the data had been labeled. A label represents the ground truth and may be as simple as a binary representation of 0 or 1. To understand the concept better this can be put into the context of the credit card example. A 1 would indicate that the transaction is fraudulent, while a 0 would indicate that it is not fraudulent. Humans often do the process of labeling data through extensive manual labor. The aforementioned example with the binary representation is a classification task. There is another task in the sub-genre supervised learning. This is regression. In regression, the task is to predict a continuous value for the given sample. In classification, the goal is to assign the sample to its correct class. In this master thesis regression is the task that is used.

In unsupervised learning labels are not used, and thus the learning not based on a ground truth. Herein, the goal is to find the underlying structure in the data, and unsupervised learning is often used for clustering and anomaly detection. If one did not have the manual labor to label the data in the fraudulent example, unsupervised learning could have been used as an anomaly detection case to try to detect fraudulent cases. Unsupervised learning is not used in this master thesis.

The last sub-genre is reinforcement learning. In this sub-genre, a reinforcement learning agent interacts with the environment and receives rewards for the perceived action. An example of this transferred over to real-life could be how a child learns to walk. First, the child starts to emulate the parents, and throughout the process, the child meets different challenges it has to overcome to finish the task. If the child received a reward for every time a couple of steps were successfully made, and no reward when the walking was unsuccessful, it would be similar to how a simplified reinforcement learning task could look. The agent receives a reward when it comes closer to the goal and no reward when the opposite happens.

Reinforcement learning is not used in this master thesis, even though its structure may be applicable to the task.

2.2.1 Missing values

Missing values imputation

Datasets often contain missing values. This means that there are no measurements done in a specific occurrence. In the example of the fraudulent credit card transaction it may be that the name of the cardholder is missing. Then the missing value is represented with a NaN. A NaN is an acronym for Not a Number, which is a way of representing a missing value.

Sensor data often have holes in their data streams as they are especially vulnerable to missing data due to practical limitations. This may be battery outages. Since many machine learning algorithms cannot handle missing values, it is necessary to impute them. Here there are several techniques. Some simple techniques which work well for time-series data are **forward fill**, **backward fill** and **rolling fill**. These techniques respectively impute the missing value with the preceding value, the succeeding value, or the mean/median of the n surrounding values. It is also possible to use techniques involving K-nearest neighbors .

Another possibility is to use other methods for imputing the missing values in the dataset. With sensors, one can use the other IoT-sensors to impute the missing values. This has shown to improve the accuracy of the imputation [23].

2.2.2 Regression

Regression is a task in machine learning where the purpose is to predict a continuous value for the given sample. This is the task that is used in this master thesis. An example of this could be a used car dealership which wants to predict the selling price of a used car based on features such as the age of the car, brand and type of transmission. A feature in machine learning is what is commonly known as a column in other applications in the world. It is information that is measurable and is used to distinguish between different samples. Error is another important term, which is the difference between the measured value and the predicted value.

To measure how well the model is doing, one uses performance metrics. There are multiple metrics used in regression tasks, but some common ones are **mean squared error (MSE)** and **mean absolute error (MAE)**.

Further in this master thesis, the metrics will be written as their acronym. In table 2.2 on the facing page, both MSE and MAE sum the mean over the difference between the predicted and the true data (e_t). The difference lies in that whilst MAE takes the mean over the absolute difference, MSE squares the errors. This makes large errors, called outliers more punishable, and is of interest in tasks where

Table 2.2: *Overview of different regression metrics, where e is the sum of the errors between the predicted value and target value.*

Metric	Formula
Mean squared error (MSE)	$\frac{1}{n}\sum_{t=1}^{n} e_t^2$
Root mean squared error (RMSE)	$\sqrt{\frac{1}{n}\sum_{t=1}^{n} e_t^2}$
Mean absolute error (MAE)	$\frac{1}{n}\sum_{t=1}^{n} \lvert e_t \rvert$

such outliers are not desirable given the context of the machine learning task. The reason for taking the square root of MSE and turning it into RMSE is to make it easier to interpret since it will come back to the original measurement unit as it was before the squaring.

2.2.3 Training process

For a machine learning model to work, one needs to specify what the goal is. A machine learning model is only as good as the data which is used to train the model. This concept is called Garbage In - Garbage Out.

In a real-world application, a machine learning model should be able to predict well on data it has never seen. This is the concept of **generalization**. When a model consistently performs well on unseen data, it is seen as generalized. It is quite easy to be blinded by a model that performs well on the data the model was trained on. The problem here lies in that the model may learn trends specific to the accessible data, which not necessarily transfers well to the unseen data. This is called **overfitting** on the training data and in such a case the generalization of the model is worsened. A model that does not perform well and only learn little or nothing about the trends in the data is said to be **underfitted**. The art of machine learning lies in finding the balance between a model that is neither underfitted nor overfitted.

To counteract both overfitting and underfitting, there is a method in data-driven modeling [24]. By splitting the data into different sets that have different functions in the machine learning task, one simulates a real-world application.

Test dataset

The first step is to take one part of the data and put it into a **test dataset**. This is illustrated in Figure 2.2 on the next page.

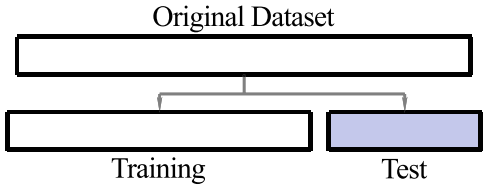


Figure 2.2: *Creating a test dataset. Adapted from [25]*

The original dataset is split into a training and test dataset. The training dataset is used to train the model whilst the test dataset should only be used to test the aforementioned generalizability of the model. The size of the splits should be such that the training data is representative of the real-world. Often, the accessible data is limited, and therefore the size of the test dataset is linked to the size of the original dataset. The larger the original dataset, the larger amount of data can be set aside for the test dataset.

To use the test dataset as a test for generalizability, a requirement is that it is used as sparingly as possible. Each use of the test data will reveal some information about the data, and **data leakage** will occur. Data leakage is the concept of data outside the training data affecting the model and therefore giving the model information it should not have. This may invalidate the model's performance since the model now has more knowledge than it should have.

Validation set

To avoid information leakage and achieve generalizability, the concept of holdout cross-validation is introduced. In this process, the training dataset is split one more time into a smaller training dataset and a validation dataset. This process can be seen in Figure 2.3.

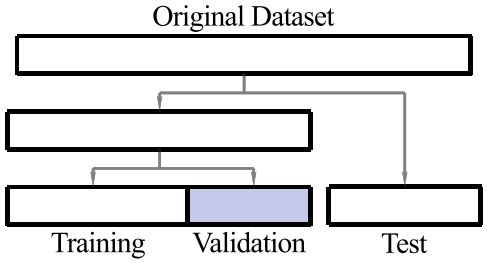


Figure 2.3: *Holdout cross-validation. Adapted from [25]*

The model is trained on the training dataset and optimized on both the training and validation dataset. The model is evaluated, and the hyperparameters are tuned on the training dataset and validation dataset. Hyperparameters are parameters of the model which are used to affect the learning process. Examples of hyperparameters

will be shown in Chapter 2.2.4. The test dataset is not used in this process and is left as the last check of the model's generalizability.

K-fold cross-validation

The challenge with using holdout cross-validation is that the data which is put in the validation dataset does not necessarily represent the distribution of the dataset as a whole. Thus the performance score on the validation dataset may be very different for different subsets of the data. This high variance leads to high uncertainty in the model's performance. There is another issue with completely removing data from the training dataset and putting it into a separate validation dataset. By removing data from the training dataset information, which may be necessary for the model to understand the trends in the data is lost. A solution to this is **K-fold cross-validation**.

Figure 2.4: *K-fold cross-validation. Adapted from [25]*

In K-fold cross-validation the data is split into k folds of data with an equal amount of data in each fold. For each iteration, a different fold is used as the validation dataset. The rest of the folds, $k - 1$ are used for training the model. This is illustrated in Figure 2.4.

2.2.4 Regression Algorithms

There are several regression algorithms that are relevant. The two main algorithms used in this study are Linear Regression and Random Forest Regression and will be described in more detail below. There are also other regression algorithms that could have been used. Neural networks can be implemented to output a continuous value, which is regression by definition. There are also variants of Linear Regression such as LASSO Regression or Ridge Regression, which have been shown to work well when dealing with variables which are collinear [26]. They impose

different types of shrinkage on the regression coefficients to avoid overfitting by avoiding extreme weights.

Linear Regression

Linear Regression is a linear algorithm, which explains the relationship between a feature and the target feature linearly. A feature is a measurable characteristic and is used as the input for the algorithm. This algorithm can also be used to explain the relationship between multiple features and the target feature. This relationship can be expressed mathematically through:

$$\hat{y} = \beta_0 + x_1\beta_1 + x_2\beta_2 + ... + x_p\beta_p \tag{2.4}$$

for i observations and k features,

$\hat{y}$ is the target value based on the dependent features,

β_0 is the y-intercept - the constant term,

x_k are the explanatory features,

β_k are the slope coefficients for every explanatory feature,

Random Forest

Random Forest is an algorithm that builds an ensemble of Decision Trees [27]. An ensemble is a collection of several individual elements, and as a musical ensemble consists of several individuals that perform as a group, the same is true for the Random Forest. As a forest is a collection of trees, the Random Forest algorithm is a collection of Decision Trees. A Decision Tree breaks down the data by asking questions which the model then uses to make its decision. It starts at its root and splits on the feature which gives the largest information gain. The information gain is a metric used to train decision trees, which determines the quality of the split. A sub-node that splits into further sub-nodes is called an internal node or decision node. A node which does not split is called a leaf node or terminal node. This basic structure of a Decision Tree can be seen in Figure 2.5.

The deeper a Decision Tree goes, the more complex the decision boundary, and the more prone it is to overfitting. This is where the Random Forest algorithm comes in. Each Decision Tree is trained on different subsets of the data, with different subsets of features. This allows the trees to learn different subpatterns. Thus, the model has a better opportunity to learn the trends in the data. The predictions of the Decision trees are averaged to give the prediction of the Random Forest model.

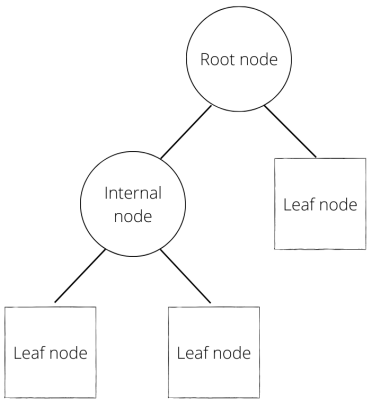


Figure 2.5: *A Decision Tree's basic structure.*

A hyperparameter is used to affect the learning process of the algorithm. For the Random Forest algorithm, there are several interesting hyperparameters:

N estimators decide the number of Decision Trees in the Random Forest. The computational cost increases and becomes more expensive as more trees are added to the model.

Min_samples_leaf describes the minimum number of samples required to be at a leaf node.

Dummy Regression

To test how the algorithms perform a dummy algorithm can be used as a baseline. A Dummy Regression is a predictor that predicts based on simple rules. An implementation of a dummy regression can be found in scikit-learn [28]. In scikit-learn the Dummy Regression can predict based on a couple of strategies. It can use either the mean or the median of the training dataset. It can also predict based on a quantile of the dataset, which also is specified as a parameter of the model. There is also the possibility to predict a constant, which is provided as an input. In this master thesis, the Dummy Regression will use mean as the predictor.

2.3 Microcontrollers

A microcontroller is a small computer on a fingernail's size, which is integrated on a data chip. The microcontroller consists of a Central Processing Unit (CPU), Random Access Memory (RAM) and persistent storage. Persistent storage is the name of a storage device that keeps its memory even after the power is shut off. The microcontroller also has a peripheral that functions as a communication tool for the outer world. The market for microcontrollers is projected to have 28.9 billion units sold in 2020, with it being estimated to grow to 38.2 billion in 2023 [29].

2.3.1 Machine learning on microcontrollers

Machine learning models integrated on microcontrollers are trained mainly on an offline basis. The model is trained on another platform, which can either be a desktop or a cloud platform. Subsequently, the model is imported unto the microcontroller. This is due to the energy expensive task of training a machine learning model and the large amount of computing power necessary in this process. Another way of implementing machine learning on a microcontroller is through online learning. In online learning, the model is continuously updated based on a stream of new data. The experimental setup of this master thesis is based on an offline learning strategy.

2.3.2 Energy states for microcontrollers

There are multiple ways to find the energy usage of a microcontroller. One of these is implementing the model on the device and then measuring the energy usage with a secondary device used for such tasks. With such a process, it is necessary to have the model ready in advance. There are also external factors that may affect energy consumption. The time needed to implement such a method is larger than the time-frame of this thesis. A second method is to approximate the device's energy usage through a formula which takes the data input as an assumption for energy consumption. This is a more straightforward method to implement and the one chosen in this thesis.

The microcontroller is assumed to have two different energy states, sleep-mode and on-mode. In sleep-mode, the energy usage is assumed to be 0. The process of going from sleep-mode to on-mode is assumed to be 0 both in the term of time and the energy cost. This mimics the process in Kraemer et al. [13]. Therefore

$$Energy\ consumption = Sampling\ rate \tag{2.5}$$

where Sampling rate $0 < x \leq 100$

2.4 Economic analysis

2.4.1 Pareto optimality

The solution to a problem is often a conjunction of multiple objectives. More often than not, these objectives conflict with each other. This is often the fact when looking at environmental and economic standpoints. Pareto optimality is given as the state where it is not possible to reallocate resources to one objective without affecting the other objectives negatively [30]. The optimal resource balance lies in the Pareto front. An illustration of this with regards to the objectives in this thesis can be seen in Figure 2.6.

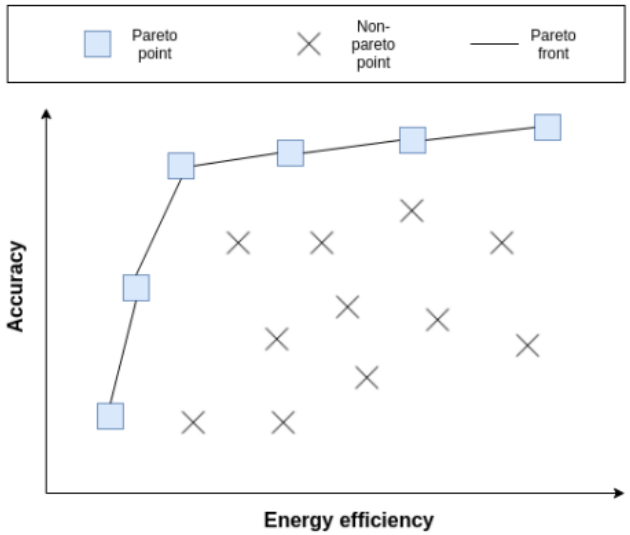


Figure 2.6: *Pareto front example. Pareto points represent the most optimal points, which will lay on the Pareto front given the specific requirements. Adapted from [31]*

2.4.2 Cost-benefit analysis

A cost-benefit analysis is a method where the drawbacks and strengths of different alternatives are ranked against each other. A systematic approach is used where each alternatives' potential benefits and costs are measured and presented to the decision-makers [32]. The costs are given in net value and have to be multiplied with a discontinuation rate. A cost-benefit analysis is often used as a macroeconomic method to understand and compare the benefits of different public measures. In Norway, all projects with a cost of over 750 million kroner need to have a cost-benefit analysis done [33].

There are different discontinuation rates. A normal one used in projects where the time-frame is below 40 years is 4 %, as shown in [34]

Chapter 3

Literature Review of Sampling strategies

Systematic literature review

A systematic literature review was used to find research that was important for this topic. This is an essential factor in research, according to [35]. The information has to be systematically identified and reported if it supports the hypothesis or not. A systematic literature review has several stages. A question is defined, data which is of relevance is searched for, then evaluated and quality assessed. Afterward, it is analyzed and combined with other data that have been previously collected. For this, Google Scholar was used as the primary search engine. The majority of the papers found were published on one of:

- Academia
- Academic Journals Database
- IEEE Xplore
- ResearchGate
- Semantic Scholar

Some of the papers used were provided by the NTNU-researchers, Frank Kraemer and Faiga Alawad personally.

Sampling strategies

Mioduszewski et al. [3] showed that the way of measuring environmental noise today, with the help of maps and temporary measurements done with expensive

recording hardware is not precise. Therefore, the idea of continuous noise monitoring, with cheap, energy-effective IoT-devices, which also have the option of replenishing their energy reserves, is of interest. To lessen the need for a secondary energy source, it is of importance to optimize the usage of energy by the IoT-device. Most environmental noise indicators do not need to be precise on the second due to the cumulative nature of these indicators, as described in 2.1.5. The key would be to lessen the amount of measurements while still maintaining an acceptable accuracy. What kind of level of accuracy that is acceptable is discussed in Chapter 2.1.2.

Sampling strategies are a tool where the measurements are done in a sampling interval, not continuously on the IoT-device. The sampling interval is the time between each measurement. By increasing the sampling interval, the IoT-device is kept longer in the sleep state, which has a lower energy usage than the recording state. Due to the lower energy consumption, the IoT-device can monitor for a more extended amount of time. This may lead to a better overview of the soundscape in the specific area. It also may lead to new, more viable business opportunities for companies working in the realm.

Previous work

Urban noise is one of the focus areas for most of the projects which focus on continuous monitoring of noise. Due to the nature of urban noisescapes, these projects are mostly focused on traffic and creating maps. Some examples of this are SONYC [5], SENSEable [6], DYNAMAP [7], CENSE [8] and Soundsensing [36].

Most of the projects where low cost is a factor use cables as a mean of power. Still, some of the projects try to use different power sources. The CENSE and SONYC projects are aiming to use solar-powered devices. Soundsensing uses a sensor network consisting of off-the-shelf sensors with embedded machine learning, which are done on-edge, to minimize the information having to be sent over the IoT-network. This means that more processing power is needed, and therefore an increased energy usage. This is one of the reasons why Soundsensing has struck up a partnership with NMBU with regard to this master thesis.

There are gaps in the discussion of sampling strategies in the aforementioned projects. Zambon et al. talk about cost in [25], but only on a macro-time scale when trying to describe the daily average sound levels of road types. Kramer et al. [13] showed the large energy savings which can be done with a static sampling rate, where reducing the sampling rate to 7 % in a 15 minutes interval did not noticeably affect the accuracy of the estimations and achieved an RMSE of 2.

There are not many mentions of an adaptive sampling rate with regards to either the accuracy of the measurements or the energy situation of the IoT-node. In other domains there are some cases of adaptive sampling rates. Bhuiyan et al. [37]

uses an event-based sampling rate, with either a low sampling rate or high sampling rate dependent on if an event has occurred or not. In the field of Human Activity Recognition (HAR), Cheng et al. [38] used a learning algorithm called Datum-Wise Frequency Selection. When looking at a combined measurement of energy cost and classification error, it performed better than the state-of-the-art algorithms. Furthermore, Trihinas et al. [39] proposed AdaM, an Adaptive Monitoring Framework for Sampling and Filtering on IoT Devices. In their paper, their adaptive monitoring framework reduced the data quantity by 74 %, had an accuracy of greater than 89 % while reducing the energy consumption by 71 % and more.

Chapter 4

Materials and Methods

The thesis' scope was to test different sampling strategies and their impact on both the accuracy and battery life on an IoT device, which records noise levels.

Due to the large amounts of preprocessing and the different techniques involved, the materials and methods chapters are put in the same chapter. The methodology used in this study consists of data preparation, an experiment setup, model selection and evaluation. Before those steps are shown and discussed, a short introduction is made of the dataset and data collection.

4.1 Data overview and collection

4.1.1 Software

This study used Python version 3.6.4 on an Anaconda platform with Numpy [40] version 1.16.2, Pandas [41] version 0.24.2, Scikit-learn [42] version 0.21.3. Matplotlib version [43] 3.0.3 and Seaborn [44] version 0.9.0.

4.1.2 Dataset

The dataset used in this master thesis consists of data collected by and at NTNU. It is used in both [12] and [13]. From this point [12] will be referred to as the NTNU master, while [13] will be referred to as the NTNU paper. The data is sensor data and provides information about the sound levels at the specific area, Koopen. The area in which the data was collected will be described in 4.1.4. The data was recorded from week 6 in 2019, until week 17 in 2019.

The dataset consist of L_{Aeq}-values, which are used as the basis for the calculations done further on in this chapter. This means that the L_{Aeq}-values will be the lowest

level in this analysis. L10 and L90 will not be used in the analysis.

4.1.3 Time interval of noise measurement

The dataset is containing data about the noisescape in a student working environment. It is crucial to choose the right time interval with regard to the problem. A time interval is an interval in which the measurements are accumulated up to. As presented in 2.1.6, 10 minutes is deemed by several sources to be a big enough time interval to capture a representative noisescape. In [12] and [13], the time interval was set to 15 minutes. By choosing 15-minute intervals as the time interval, it has a balance of being precise enough that a student may check for the interval each 15 minute, and not too granular, so that it has too much information for a student looking for a silent place to study. This experimental setup is going to use the same time interval.

4.1.4 Data collection

The data was collected in Koopen, a study area at NTNU used by the students at the program Electronic engineering. It is meant to be used as a meeting place for students at the study program, and as a place for group work [45]. Thus the students need to know how noisy it is at Koopen before going there, as noise has a negative impact on studying [18].

The data collection setup consisted of five sensors. They were connected to a router through WiFi-connection. These then send the data to a database on a server. This is illustrated in Figure 4.1 on the facing page.

The calibrated range of the sensors is between 50 dB and 100 dB [12]. The sensors which recorded the sound level were from Libelium, a manufacturer of IoT-devices. The devices used consist of two parts; one is a Libelium Waspmote Plug & Sense! Smart Cities Pro and the other is a Libelium Noise Level Sensor. The devices were placed 2.5 meters above the ground due to the placement of whiteboards on the walls. The devices were connected to a power supply and, thus, could record noise levels continuously throughout the data collection period. The data was sent to a database through a Cisco router, as pictured in Figure 4.1 on the next page. The setup of the Libelium devices can be pictured in Figure 4.2 on the facing page.

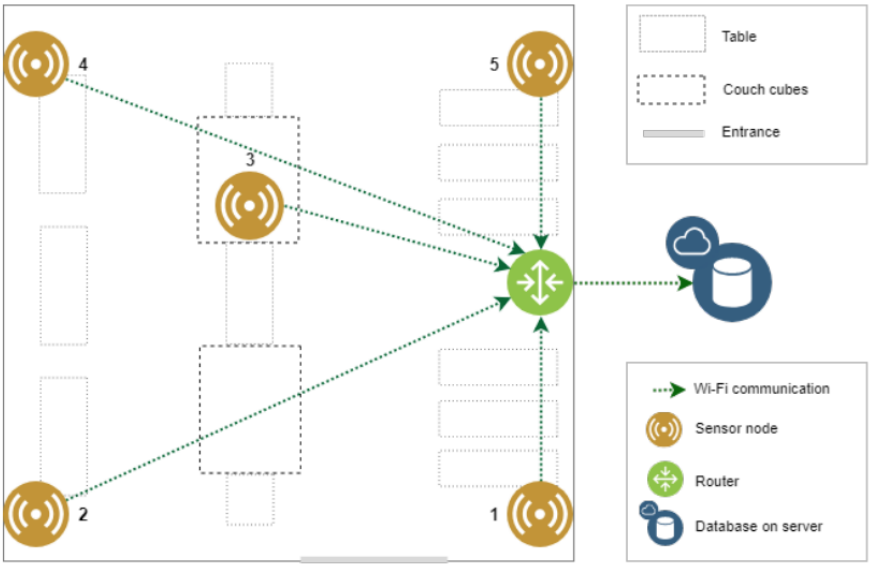


Figure 4.1: *Setup of sensor nodes in Koopen. Figure adapted from [12].*

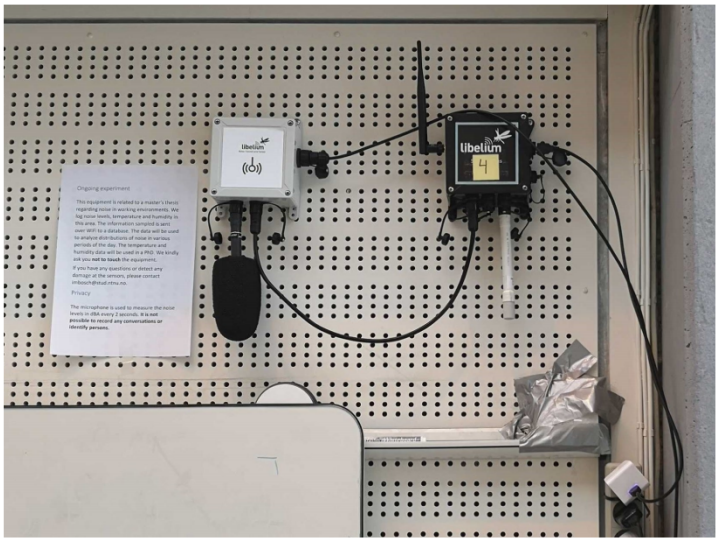


Figure 4.2: *Waspnodes setup in Koopen. The Libelium Waspmote Plug & Sense! Smart Cities Pro on the right and the Libelium Noise Level Sensor on the left. Figure adapted from [12].*

,

4.2 Data preprocessing

A-weighting was used as a filter for the dataset. The process of A-weighting is described in 2.1.3 on page 7.

4.2.1 Calibration and time-shifting

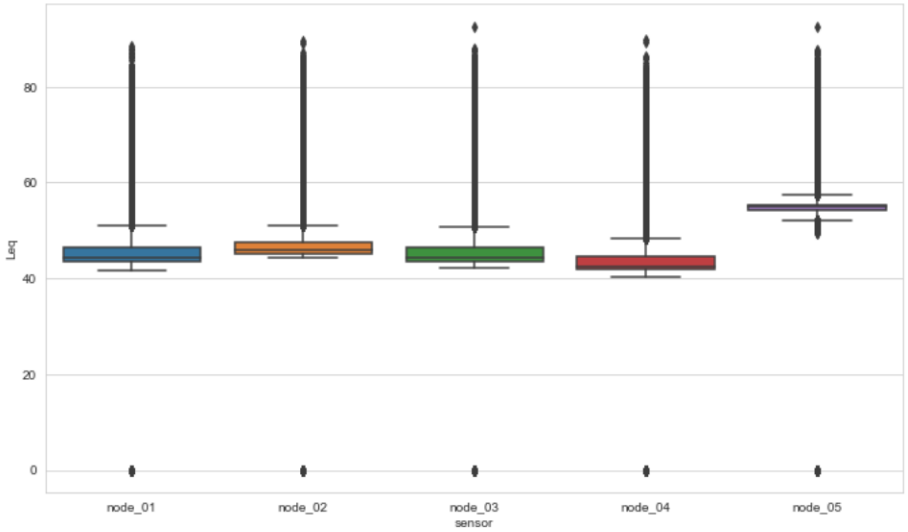


Figure 4.3: *Boxplot of L_{Aeq} values of all sensors.*

Even though the Libelium devices are calibrated in a range of 50 to 100 dBA Figure 4.3 shows that there is a multitude of measurements outside this range. There seems to be a lower limit of 40 L_{Aeq}. From the NTNU-master, it is stated that it is possible for the Libelium devices to do lower readings than the stated calibrated range, but that the uncertainty in these readings are greater than in the calibrated range. The noise level recordings are made every two seconds, which is the highest reading frequency that the devices are capable of.

Sometimes in the dataset, the different sensor measurements are misaligned by a second. Since a 15-minute interval is used for the evaluation it is possible to shift the misaligned measurements back a second so that all measurements are on the same time interval of 0, 2, 4, 6, 8 seconds. The bigger noise picture of the 15-minute interval will still be maintained, and it is not a significant difference that some measurements were shifted a second. In the places where measurements are made in the second 0 and in the second 1, the measurements made on even-numbered time index were kept.

4.2.2 Data statistics

To fully grasp how the dataset is constructed, one needs to go deeper into each sensor. A simplification made in this master thesis is to use only one sensor as the basis for prediction, even though in theory, one could try to predict values for all five sensors. Another possible route would be to use the other sensors' data as different test sets to see if the model is generalizable to other sensors. One would have to make sure that there was no data leakage in that the time frame, which is predicted in the other sensors, does not overlap with any of the training or validation data.

Table 4.1: *Statistics of L_{Aeq} values of all sensors.*

sensor	count	mean	std	min	25%	50%	75%	max
node01	3034904	45.5	7.9	0.0	43.5	44.3	46.5	88.4
node02	3104532	47.5	4.0	0.0	45.2	45.8	47.5	89.6
node03	3158160	46.3	4.9	0.0	43.5	44.3	46.4	92.6
node04	3159952	44.6	4.8	0.0	42.0	42.5	44.5	89.7
node05	3050684	55.1	3.1	0.0	54.1	55.1	55.4	92.4

One can observe in Table 4.1 that sensor node 05 has higher values than the other four sensor nodes. This is regarding the mean-value, which is 9.2 (L_{Aeq}) higher than sensor node 03, and also around that range for the rest of the sensors. This is also true regarding the 25, 50 and 75 quantiles, where sensor node 05 is 8-13 L_{Aeq} higher than the rest of the respective quantiles for the rest of the sensors.

When looking at Figure 4.1 on page 27, one may observe that sensor node 05 is placed over tables that function as group working places, which may explain some of the higher values on that sensor node, but then one would assume that sensor node 01 also had higher measurements than the rest. Since this is not the case, it is unknown why this is so, but an assumption is made that there is a difference in the calibration of the sensors.

Sensor node 03 was concluded to be the most stable in Bosch [12]. In Figure 4.4, the $L_{Aeq,15}$ for the observation period can be observed. One can clearly see the period in which students are present in Koopen. This is from 06:00 till 19:00, with some outliers as well in e.g. weeks 12 and 13. Friday afternoons are quieter than other weekdays. Weekends and holidays (week 16) are quieter than the weekdays. In the nighttime, the working space is the quietest. There are two nights which are outliers in week 12, which may be because of late-night studying or a party.

Figure 4.4: *$L_{Aeq,15}$ of Sensor node 03 for the observation period.*

4.2.3 Missing values

If there are missing values in the measurements, it will affect the experiment. This is due to how the experimental setup has been designed. A mask will walk through every 15-minute window and pick out measurements based on a sampling percentage. For an experimental setup like this, there should be 450 measurements in each 15-minute window.

In the NTNU-paper, only one sensor was used, sensor node 01, and all 15-minute windows with missing data were discarded. To get an overview of the amount of missing data which has to be discarded or how many missing values has to be imputed a deeper analysis is performed.

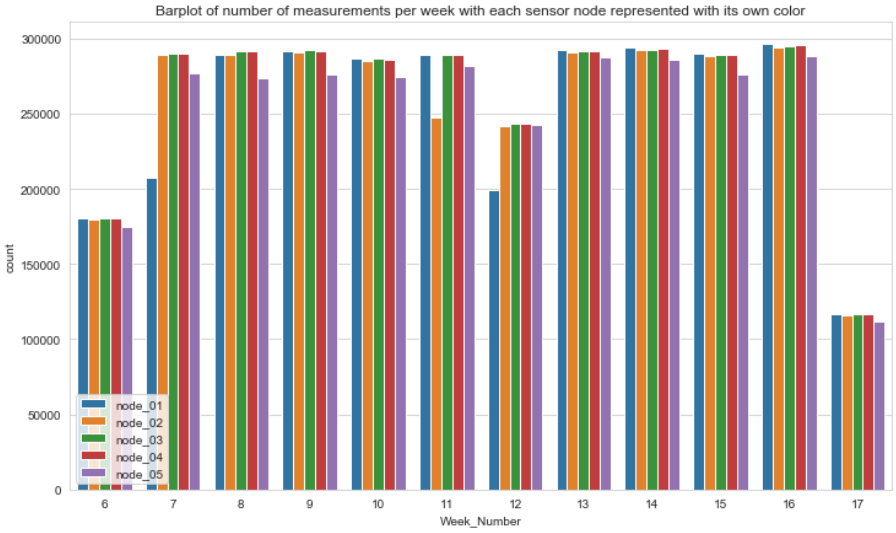


Figure 4.5: *Barplot of the number of measurements per week with each sensor node represented with its own color. The number of measurements on the y-axis. X-axis represented by week.*

The data was collected over 11 weeks. Due to the noisy environment and observations based on the calculated statistics on the sensor nodes, an assumption is made that all L_{Aeq}-values with a value 0 are wrong measurements and are set to NaN. There were 56 336 (1.9%) L_{Aeq}-measurements with value 0 in sensor node 01, 53 in sensor node 02, 52 in sensor node 03, 72 in sensor node 04 and 3418 in sensor node 05.

By looking at Figure 4.5, one can see that Week 6 and Week 17 have fewer measurements than all other weeks. This may indicate that the data collection was started and ended midway through a week. Furthermore, node 01 and node 05 had approximately 900 000 (3.9 %) and 750 000 (3.5%) fewer measurements than the other three sensors, as shown in Table 4.1 on page 29. When all sensors are

functioning properly, node 05 seems always to have a little fewer measurements than the other sensor nodes. In week 12 the sensors nodes had approximately a 15 % reduction overall in measurements in comparison to both the week before and after.

Frequency of missing measurements in the 15-minute windows

Each 15-minute window in the dataset will be examined for the number of missing measurements. This is important because if there are any specific time intervals with a large number of missing measurements, these may be necessary to either impute, or, if the percentage of the missing measurements is abnormally high, that specific time interval may be discarded. By discarding data, the model's ability to discern the trends may be weakened, yet if the specific time interval has a large percentage of missing measurements, it may be necessary.

By counting the number of measurements in each 15-minute window one can get an overview of the quality of the dataset. If the majority of the 15-minute windows have more than 50 % missing measurements, it may be difficult to conduct the experiment with the masking method mentioned, and another method has to be introduced. In Figure 4.6 on the next page, each sensor node with it's associated 15-minute windows, and their number of missing measurements is presented. From the analysis performed, it looks like not one 15-minute window has measurements for all 450 sample points and that the lowest amount of missing measurements in a 15-minute window is 6.

There are two areas on the figure which show a large amount of count of instances. This is from 0-50 missing measurements in a 15-minute window and, on the very end, at 450 missing measurements. In the other areas, the trend is some small instances with missing measurements, but not any systematic trends, as seen in the two aforementioned areas. Sensor node 01 is the lone outlier, and it has several windows with around 300 missing measurements.

The large increase in instances of windows with 100 % of missing measurements can be assumed to be partly due to the aforementioned weeks 6 and 17, where the data collection did not start until midweek. When focusing on the other area of interest, the interval between 0 and 50, there seems to be cohesion between the sensors. In Figure 4.7 on the facing page, this area is presented. The plot shows a clear trend where the number of missing measurements are following the same tendencies with the most abundant instances of missing measurements being in the range of 10 to 20 per 15-minute window.

The findings lead to the decision to impute the missing values. The NTNU-paper discarded all 15-minute windows with missing values, yet the results in that paper could not be reproduced by the author of this master thesis, and zero 15-minute windows without missing values were found.

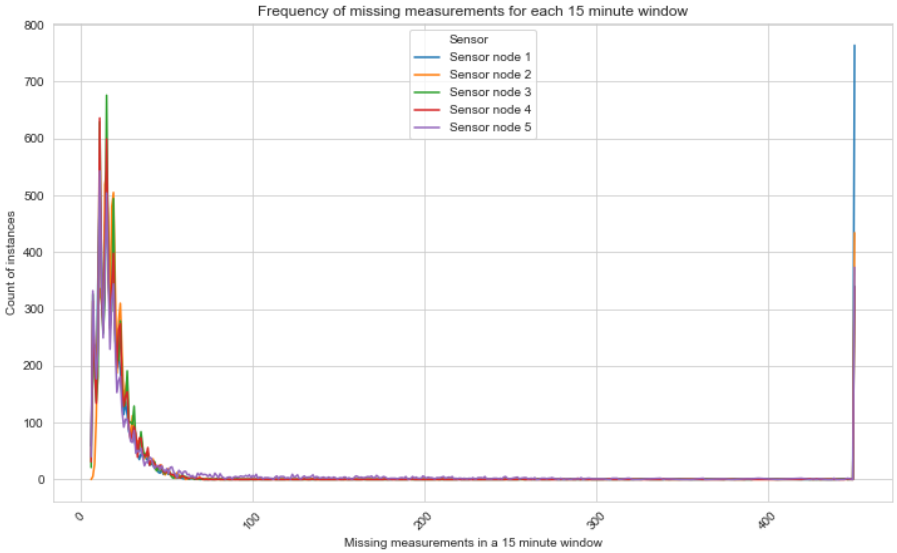


Figure 4.6: *Frequency of missing measurements for each 15-minute window. Count of instances on the y-axis and the number of missing measurements in a 15-minute window on the x-axis.*

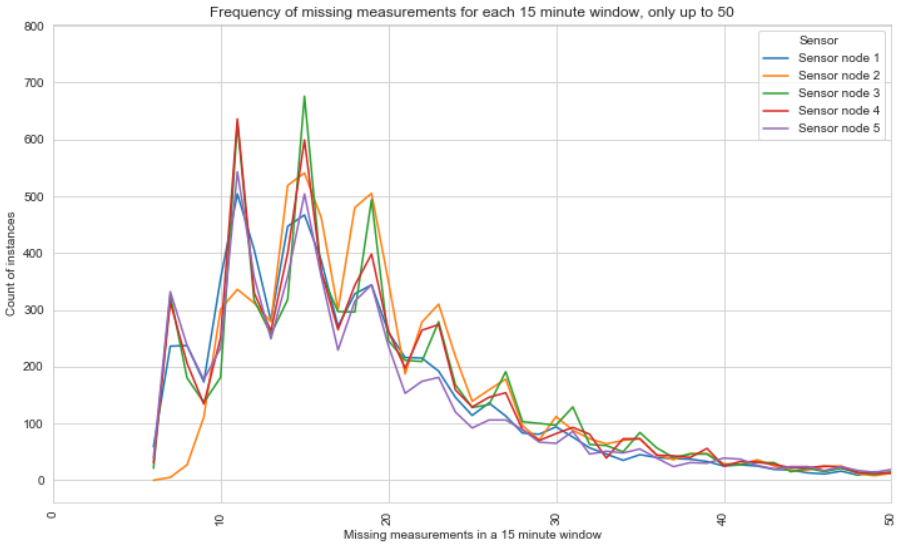


Figure 4.7: *Frequency of missing measurements for each 15-minute window. Count of instances on the y-axis and the number of missing measurements in a 15-minute window on the x-axis. The only difference to the previous figure is that the number of missing measurements are limited to 50 on the x-axis.*

Process of the imputation of missing values

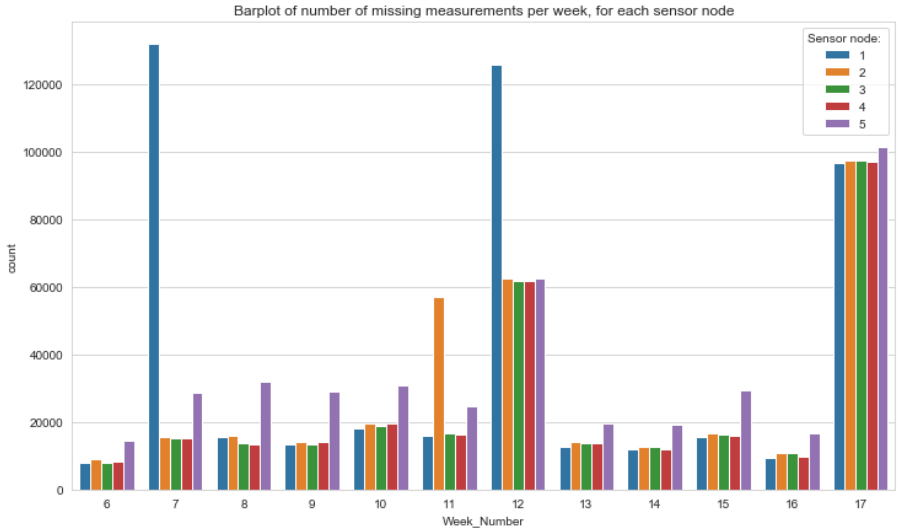


Figure 4.8: *Barplot of the number of missing measurements per week, for each sensor node. The number of measurements on the y-axis. X-axis represented by week.*

Due to the large number of missing values discovered, with not one 15-minute window without missing values, a decision is made to impute the missing values. The data from week 16 is chosen as test data due to the full week that is represented. It is the last week with a low degree of missing measurements, as seen in Figure 4.8.

Sensor node 03 is chosen to be the main sensor node. It will from now on be referred to as the master sensor node. Along with sensor node 04, it is the sensor with the lowest amount of missing measurements overall, as seen in Figure 4.8.Another important factor as to why sensor node 03 is chosen as the master sensor node is its central placement in the room, as seen in Figure 4.1 on page 27. Isolated, the sensor's placement is not that important, but if one wishes to use the other sensor nodes to impute the missing measurements in the master sensor node, it will affect the imputed values. The reasoning is that the central placement of sensor node 03 in the room will give it a more correct soundscape than if sensor node 04, which is placed in the corner, was chosen.

The imputation method is as follows:

For the measurements where sensor node 03 does not have a value, an average over the other sensor nodes' measurements for that specific missing time point is imputed in sensor node 03. The number of sensor nodes with measurements varies and may be in the range of 0 to 5. In the cases where no measurements have been

made, a normal forward imputation is made where the missing value is replaced with the value of the previous measurement. In the cases where sensor node 03 does not have a missing measurement, the existing measurement is kept.

Sensor node 05 has different average values than the other sensor nodes, as seen in Table 4.1 on page 29, and it is presumed that this is due to a different calibration. To not skew the imputation method, all data in sensor node 05 is re-scaled so that the mean of sensor node 05 is equal to the mean of sensor node 03. This process is done by multiplying each measurement for sensor node 05 with the constant from Equation 4.1. This is done on the dataset where week 16 is excluded and all measurements with noise level value 0 are replaced with NaN. The mean of sensor node 03 is 46.94, while the mean of sensor node 05 is 55.20.

$$ReScalingConstant = \frac{sensornode03_{mean}}{sensornode05_{mean}} \tag{4.1}$$

From this point in the thesis, the data for week 17 is discarded due to the large degree of missing measurements. A comparison between the L_{Aeq}-values of sensor node 03 before and after the imputation is provided in Figure 4.9. The sensor data have a wider range of values after the imputation, but otherwise, the distribution does not change much.

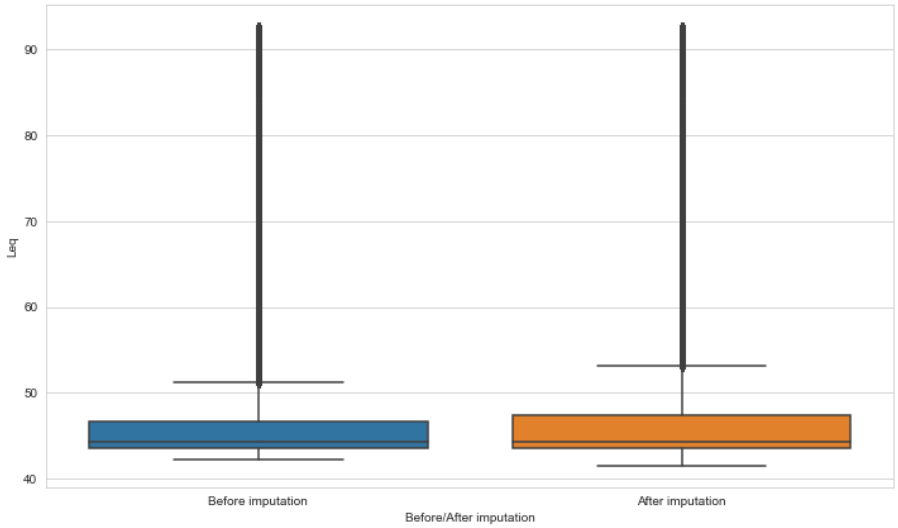


Figure 4.9: *Comparison of L_{Aeq}-values for Sensor node 03 before and after imputation.*

4.2.4 Masking

To decrease the energy usage, a subsampling strategy will be implemented on the data. This will lead to the device being on for less time, and thus the energy usage decreases. This also leads to fewer recorded measurements, yet one does not wish this to affect the accuracy such that it crosses the threshold of ±3 L_{Aeq}, defined in Chapter 2.1.2.

When subsampling from the original dataset, a mask will be applied to every 15 window to pick out measurements based on a sampling percentage. The mask is a one-dimensional NumPy array with 450 elements. The mask elements which are marked with a 1 are the places where measurement is picked out; the rest is discarded. One mask is created for each subsampling rate and then copied the number of 15-minute windows in the dataset times. Thus a mask which covers the whole dataset is created.

The RMSE is calculated between the L_{Aeq}-value for the original 15-minute window and the L_{Aeq}-value for the subsampled 15-minute window. Then the mean of the RMSE across all 15-minute windows based on a specific subsampling percentage is calculated. This is done to understand how each subsampling performed. RMSE will then be a measure of how representative the subsampled measurements are of the true variation in sound.

Masking methods

Two different masking methods are implemented and tested. Masking method 1 (MM1) picks out measurements based on set intervals. As an example, when using a subsampling percentage of 50 % MM1 marks every other sample with a 1, and the sample is copied into the subsampled data frame. The relationship between subsampling percentage and the frequency can be seen in Table 4.2 on the facing page. MM1 is limited to whole numbers, which can be multiplied into 450.

Masking method 2 (MM2) uses a NumPy random function, NumPy.random.choice, to create a one-dimensional NumPy array with length 450, filled with zeros and ones where the ratio of ones and zero is determined by the subsampling percentage. Thus, the masking is applied at random. With a masking method that uses a random selection, there is no set frequency. This means that it is possible to use all possible subsampling percentages in the range of 0 to 100. The distribution of elements picked out by the masking method across the binned elements can be seen in Figure 4.10. What can be seen in this figure is that there are some bins which are a bit over-represented in comparison to the neighboring bins. This can be seen for the subsampling rate of 35 % and bin 271-315 and the neighboring bins. This does not seem to be a problem.

Table 4.2: *Overview of the relationship between subsampling percentage, number of samples and frequency of subsampling with MM1.*

Subsampling percentage	Number of samples	Frequency
100 %	450	1
50 %	225	2
20 %	90	5
10 %	45	10
6.7 %	30	15
3.3 %	15	30
2 %	9	50
1 %	4.5	100

Selection of the masking method

The two masking methods are applied to the dataset and compared. It is essential to see how they fare in the different subsampling areas, as there may be areas where one has a better performance than the other. In Figure 4.11 on page 39, MM1 (called Determined in the figure) works better when the subsampling rate is over 10 %. When the subsampling rate is below this threshold, MM1 does not perform much differently from MM2 (called Random in the figure).

The error bands on the figure show the variation dependent on the week. MM1, Determined, is more stable, than MM2, Random, which has more variation between the weeks. However, despite the difference in variation, MM2 provides much greater flexibility in usage, with the ability to use a wide range of subsampling percentages. This is due to it not being constrained by the frequency of the set masking method, where each sample has to come every 2 or 4 sample. This flexibility is determined to outweigh the lower performance in subsampling percentages over 20 % and the larger variation in performance between the different weeks.

The subsampling percentages that MM2 will use are from 100 % to 5 %, with a decrease of 5 % between each subsampling percentage.

	1-45	46-90	91-135	136-180	181-225	226-270	271-315	316-360	361-405	406-450
100%	45	45	45	45	45	45	45	45	45	45
95%	43	41	44	44	44	44	43	43	45	42
90%	40	41	43	41	41	42	41	40	42	40
85%	37	32	36	36	38	37	35	35	34	36
80%	37	36	31	35	34	39	38	40	43	37
75%	28	37	37	36	33	42	31	38	34	35
70%	27	31	33	30	35	35	31	27	31	27
65%	32	36	27	30	27	23	32	29	28	29
60%	30	31	29	25	22	20	22	27	26	25
55%	25	29	20	27	23	23	31	26	32	33
50%	25	21	24	24	15	21	20	22	24	23
45%	13	20	23	15	20	20	17	21	17	19
40%	15	13	11	22	15	13	21	14	15	15
35%	12	13	16	14	14	13	20	15	19	17
30%	12	15	21	13	13	12	17	17	12	14
25%	9	15	17	10	12	11	10	7	15	16
20%	7	8	9	10	3	8	8	15	12	5
15%	9	9	3	5	8	4	10	3	8	13
10%	3	2	2	5	5	5	5	6	4	6
5%	2	5	3	2	2	1	4	4	0	3

Figure 4.10: *An overview of MM2 distribution across binned values of a window with 450 elements. Rows indicate sampling percentage, and the column are the bins that the elements are divided into.*

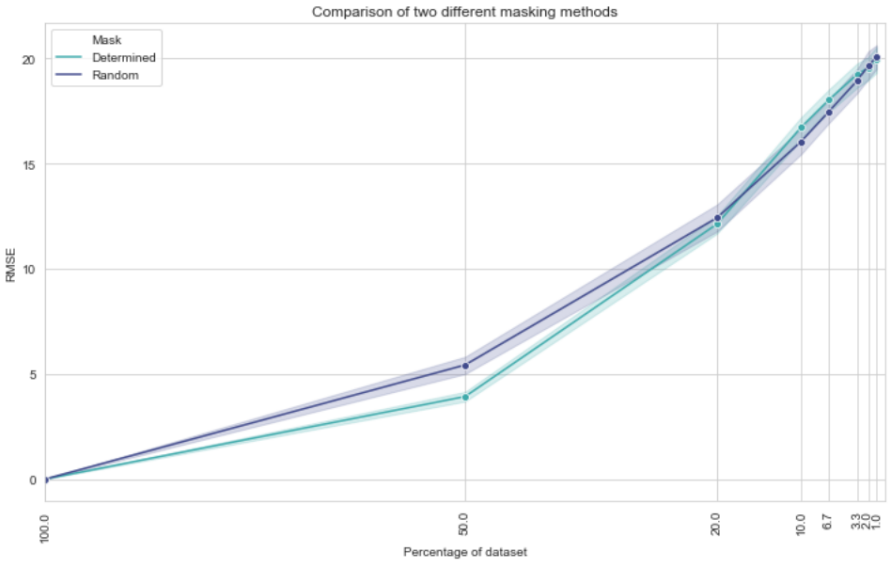


Figure 4.11: *Comparison of two masking methods with specific week. RMSE on the y-axis and percentage of dataset (subsampling rate) on x-axis.*

4.2.5 Summary of processes and methods implemented

Week 17 was discarded due to the large number of missing measurements. Week 16 was separated into a test dataset.

Sensor node 03 was chosen as the master sensor. Sensor node 05 was re-scaled with regards to the mean of sensor node 03. Then a weighted mean between all sensor nodes was applied on rows with missing measurements. MM2, Random, was chosen as the masking method due to the flexibility this method has. The subsampling percentages that MM2 will use are from 100 % to 5 %, with a decrease of 5 % between each subsampling percentage.

The resulting dataset comprimises a training dataset with 2 908 350 rows and a test dataset with 302 400 rows.

The dataset consists of an index that has an interval of 2 seconds and a column with L_{Aeq}-values for the specific timestamp.

4.3 Experimental setup

An experiment has to be set up to test the hypothesis. The hypothesis is how often the IoT-device needs to record noise levels while keeping the accuracy below the error threshold regarding the energy-accuracy trade-off. This hypothesis is tested by seeing if it is possible to predict the RMSE of the L_{Aeq}-values with the features available in the dataset. The data which is going to be used is the preprocessed data of the master sensor, sensor node 03. This data frame has a time index and a column with L_{Aeq}-values, and the rows have a measurement every 2 seconds. The data frame will be called **O.G-Dataframe**, an abbreviation for Original-Dataframe, and can be seen in Figure 4.12.

	Leq
time	
2019-02-06 16:15:00+00:00	46.5302
2019-02-06 16:15:02+00:00	48.6518
2019-02-06 16:15:04+00:00	50.2161
2019-02-06 16:15:06+00:00	47.0890
2019-02-06 16:15:08+00:00	50.6675
2019-02-06 16:15:10+00:00	52.5284
2019-02-06 16:15:12+00:00	54.9720
2019-02-06 16:15:14+00:00	49.8727
2019-02-06 16:15:16+00:00	49.9323
2019-02-06 16:15:18+00:00	50.5314

Figure 4.12: *Original-Dataframe (O.G-Dataframe) containing L_{Aeq} of the master sensor. Time-index with even spacing of 2 seconds and the accompanying L_{Aeq}-value.*

The experiment should mimic a real-life scenario where one does not have information about the next measurements. The goal is to test how different subsampling rates affect the error in the predictions made by the machine learning model. To test the different subsampling rates, the original data frame has to be transformed into the time interval, which was decided in Chapter 4.1.2 on page 25, 15 minutes. A major question that the experiment will try to answer is if it is possible to predict the error of window *n+1*, while only using the info of the previous window, *n* as input into the already trained machine learning model.

The subsampling space used will be every subsampling rate from 100 % to 5 %, with an interval of 5 %. This gives 20 different subsampling rates. For each subsampling rate MM2 will pick out the measurements and put these in a new, empty

data frame. Simultaneously the feature extraction will be performed. This process will be described more in Chapter 4.3.3.

All subsampled data frames together will be called **SubX-Dataframes**. If there is talk about a specific subsampling rate, the X will be replaced with the subsampling rate, e.g., Sub20-Dataframe. If a subsampling rate is not mentioned, X indicates all subsampling rates. An excerpt of the Sub20-Dataframe is shown in Figure 4.13.

time	max	min	max-min	mean	std	std/mean	skew	Leq_orig	Leq_sampled	RMSE	Subsampling rate
2019-02-06 16:15:00+00:00	54.82	45.35	9.47	48.54	2.07	0.04	0.77	49.33	49.08	0.25	20
2019-02-06 16:30:00+00:00	58.90	45.03	13.88	48.47	2.50	0.05	1.46	65.94	49.43	16.51	20
2019-02-06 16:45:00+00:00	56.09	45.51	10.58	49.00	2.47	0.05	0.94	50.38	49.81	0.56	20
2019-02-06 17:00:00+00:00	92.54	45.01	47.53	49.24	5.45	0.11	6.17	66.11	73.27	7.16	20
2019-02-06 17:15:00+00:00	53.93	44.55	9.38	46.74	2.23	0.05	1.56	69.61	47.46	22.15	20
2019-02-06 17:30:00+00:00	57.37	44.48	12.89	46.84	2.35	0.05	1.99	68.24	47.77	20.47	20
2019-02-06 17:45:00+00:00	66.01	44.28	21.74	47.04	2.92	0.06	3.67	65.63	50.05	15.58	20

Figure 4.13: *Sub20-Dataframe containing time-index with even spacing of 15 minutes and the accompanying features derived from the L_{Aeq}-values in the particular 15-minute window. Leq_orig and Leq_sampled are dropped from the data frame but kept in the figure to show the RMSE-calculation. Rounding is performed for clarity in the figure, but not in the analysis.*

All subsampling rates within the SubX-Dataframes will be copied into one large data frame. The data frame, called **15Min-Dataframe**, is the dataset in which predictions are going to be made on. It has a row for every 15 minutes, which indicates every 15-minute window, *n, n+1, n+2, n+3,... n+k*. These *n, n+1, n+2, n+3,... n+k* windows exist for each subsampling rate, which means that each time window *n* has 20 instances in the 15Min-Dataframe, with different values. An example of this can be seen in Figure 4.14.

time	max	min	max-min	mean	std	std/mean	skew	Leq_orig	Leq_sampled	RMSE	Subsampling rate
2019-02-06 16:15:00+00:00	56.47	45.35	11.12	48.66	2.25	0.05	0.88	49.33	49.33	0.00	100
2019-02-06 16:15:00+00:00	52.97	45.37	7.60	48.12	1.94	0.04	0.69	49.33	48.58	0.75	5
2019-02-06 16:15:00+00:00	54.86	45.37	9.49	48.54	1.93	0.04	0.76	49.33	49.02	0.31	25
2019-02-06 16:15:00+00:00	56.47	45.47	11.00	48.63	2.58	0.05	0.98	49.33	49.52	0.19	15
2019-02-06 16:15:00+00:00	56.47	45.37	11.10	48.68	2.26	0.05	0.90	49.33	49.36	0.03	85
2019-02-06 16:15:00+00:00	55.70	45.37	10.33	48.84	2.22	0.05	0.64	49.33	49.46	0.13	30
2019-02-06 16:15:00+00:00	56.47	45.35	11.12	48.65	2.23	0.05	0.90	49.33	49.31	0.02	80
2019-02-06 16:15:00+00:00	55.65	45.37	10.27	48.62	2.20	0.05	0.80	49.33	49.24	0.09	75
2019-02-06 16:15:00+00:00	54.82	45.35	9.47	48.54	2.07	0.04	0.77	49.33	49.08	0.25	20
2019-02-06 16:15:00+00:00	56.47	45.35	11.12	48.65	2.25	0.05	0.88	49.33	49.32	0.01	95
2019-02-06 16:15:00+00:00	56.47	45.35	11.12	48.65	2.24	0.05	0.93	49.33	49.32	0.01	40
2019-02-06 16:15:00+00:00	55.70	45.35	10.35	48.73	2.24	0.05	0.82	49.33	49.39	0.05	70
2019-02-06 16:15:00+00:00	55.70	45.37	10.33	48.63	2.40	0.05	0.94	49.33	49.40	0.07	45
2019-02-06 16:15:00+00:00	56.47	45.35	11.12	48.60	2.27	0.05	1.02	49.33	49.29	0.04	50
2019-02-06 16:15:00+00:00	55.65	45.35	10.30	48.62	2.19	0.05	0.76	49.33	49.24	0.10	65
2019-02-06 16:15:00+00:00	55.65	45.35	10.30	48.64	2.24	0.05	0.80	49.33	49.30	0.04	55
2019-02-06 16:15:00+00:00	56.47	45.35	11.12	48.75	2.34	0.05	0.77	49.33	49.46	0.13	60
2019-02-06 16:15:00+00:00	53.06	45.47	7.59	48.56	2.22	0.05	0.68	49.33	49.17	0.17	10
2019-02-06 16:15:00+00:00	55.70	45.35	10.35	48.66	2.21	0.05	0.78	49.33	49.29	0.04	90
2019-02-06 16:15:00+00:00	56.47	45.58	10.89	48.41	2.10	0.04	0.97	49.33	49.01	0.33	35
2019-02-06 16:30:00+00:00	58.90	45.03	13.88	48.47	2.50	0.05	1.46	65.94	49.43	16.51	20
2019-02-06 16:30:00+00:00	63.88	45.07	18.81	48.33	2.78	0.06	2.95	65.94	50.31	15.63	15

Figure 4.14: *15min-Dataframe containing time-index with all 20 instances of same 15-minute window from different SubX-Dataframes shown with accompanying features + 2 extra instances of next 15-minute window. Leq_orig and Leq_sampled are dropped from the data frame but kept in the figure to show the RMSE-calculation. Rounding is performed for clarity in the figure, but not in the analysis.*

4.3.1 Target variable

The target variable in the 15Min-Dataframe is the feature RMSE. The RMSE is the difference between Leq_orig and Leq_sampled of the 15-minute window. Leq_orig is the average L_{Aeq}-value of the 15-minute window without any subsampling performed. In other words, it is the average L_{Aeq}-value of the Sub100-Dataframe. This can be seen on Figure 4.14 on the row with value 100 in the Subsampling rate. Leq_sampled is the average L_{Aeq}-value of the 15-minute window with the specific subsampling implemented. The RMSE is calculated for all SubX-Dataframes for each 15-minute window, *n, n+1, n+2, n+3,... n+k*.

Leq_orig and Leq_sampled is calculated through the formula in Equation 2.3 on page 9. The reason for using the average L_{Aeq}-value from the Sub100-Dataframe as the ground truth for average L_{Aeq}-values is that this particular subsampled dataframe has all 450 measurements included.

A column of RMSE-values for all 15-minute windows is added to the 15Min-Dataframe, as seen in Figure 4.14, and this will be the target variable. The target variable will be a continuous value which cannot be lower than 0 and higher than the difference between the highest and lowest value in each 15-minute window.

4.3.2 Distribution of errors (RMSE)

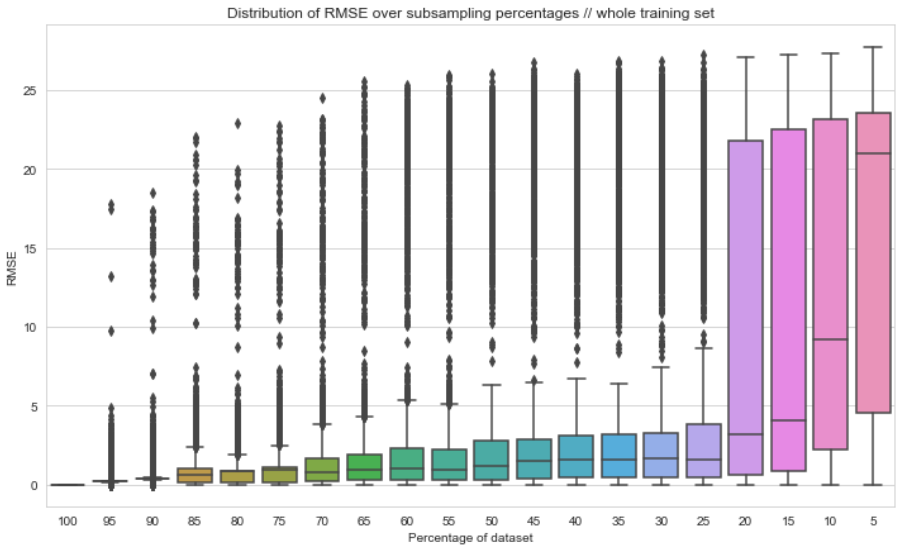


Figure 4.15: *Distribution of RMSE (L_{Aeq}) over subsampling percentages for the whole training set. 100 % indicates that the full dataset has been used.*

An overview of the error distribution for all subsampling percentages is shown in Figure 4.15 on the preceding page. The fewer measurements used in the time interval, the higher the RMSE. This increase is increasing steadily up to the subsampling percentage of 25 %. From a subsampling percentage of 20 %, there is a substantial increase in errors over 10 RMSE. The challenge will be to train a model that can model the error distribution for all subsampling percentages.

4.3.3 Feature extraction

In the O.G-Dataframe, there are two columns that can be used to generate more features to predict the target variable. This is either the datetimeindex or the column with L_{Aeq}-values. It is of interest to test if a time-independent dataset has the potential to predict the RMSE. Therefore, preliminary, only the column with L_{Aeq}-values is used to generate more features. Statistics for each subsampled 15-minute window is calculated based on the L_{Aeq}-column. The statistics were calculated using pandas.DataFrame.resample and the time interval was set to 15 minutes. The statistics calculated were max, min, standard deviation, skewness and mean. Max-min and std/mean are also added as features after doing row-wise operations on the dataset.

These values are calculated for the 15-minute window for each subsampling rate and added to the associated SubX-Dataframe, as shown in Figure 4.13. The subsampling percentage is also added as a feature to each SubX-Dataframe. Then every SubX-Dataframe is added to the 15Min-Dataframe, which can be seen in Figure 4.14 on page 43.

4.3.4 Time-dependent features

Another variant of the experiment is done with two additional features, hour and day of week. These two features are included in the model. Day of week is a feature with a range of integers from 0 to 6, while hour is a feature with a range of integers from 0 to 23. The features are the two most important features used in the NTNU-paper to predict their noise predictors. They are added to the 15Min-Dataframe with the other features in a separate experiment before predicting, and no other changes are made.

4.3.5 Set selection

Week 16 is set aside as a test set. To train the model and evaluate the performance of the models, a K-fold cross-validation is implemented. Each week represents a different fold. With week 16 set aside as test set, ten weeks left for the training and evaluation process. These weeks are 6, 7, 8, 9, 10, 11, 12, 13, 14 and 15.

The folds are paired through a random selection. The resulting distribution of folds towards the training and validation set can be seen in Table 4.3.

Table 4.3: *Overview of how the different weeks are split into training and validation sets in the K-fold cross-validation.*

Fold	Training set (weeks)	Validation set (weeks)
1	6, 7, 8, 9, 11, 13, 14, 15	10, 12
2	6, 8, 9, 10, 12, 13, 14, 15	7, 11
3	7, 8, 10, 11, 12, 13, 14, 15	6, 9
4	6, 7, 9, 10, 11, 12, 13, 15	8, 14
5	6, 7, 8, 9, 10, 11, 12, 14	13, 15

This results in 5 different models, which are evaluated on different subsets of the data. The variation in scores between the different models will be used to determine the stability of the models.

4.3.6 Metrics

The metric used to determine the accuracy of the model will be RMSE of RMSE. The target variable (RMSE) is calculated between the average L_{Aeq}-value of the Sub100-Dataframe and the average L_{Aeq}-value of each SubX-Dataframe for each 15-minute window, *n, n+1, n+2, n+3,... n+k*. After the process of predicting the RMSE, one gets RMSE of RMSE. This says how many $L_{Aeq,15min}$ the prediction is from predicting the correct RMSE of the 15-minute window.

4.3.7 Using window *n* to predict RMSE of window *n* (Baseline)

Before predicting the RMSE of window *n+1* based on the features of the *n* window, a baseline is created. The baseline is found through predicting the RMSE of window *n* based on the statistics of the same window, *n*.

Three different algorithms are used; a Linear regression, Random Forest Regression and Dummy Regression. The set selection is shown in Table 4.3. The Dummy Regression uses the mean of the target variable in the whole training set as a predictor. The Linear Regression is trained out of the box with no hyperparameters tuned. The Random Forest Regression has the parameters n_estimators set to 10 and min_samples_leaf to 0.001. These values were found through a little grid search, which consisted of training several models with different values for the parameters. The goal of the hyperparameter-tuning is both to prevent overfit towards the training dataset and reduce the bias. The values tested were 10, 20 and 30 for the n_estimators. In addition, the min_samples_leaf was tested on 0.1, 0.01 and 0.001. This resulted in nine models. The model with n_estimators set to 10 and min_samples_leaf to 0.001 was the best performing. An overview of the Random Forest models and the two hyper-parameters tested is presented in Table 4.4.

Table 4.4: *Overview of all Random Forest Regression models based on different combinations of hyperparameters.*

Model	n_estimators	min_samples_leaf
1	10	0.1
2	10	0.01
3	10	0.001
4	20	0.01
5	20	0.01
6	20	0.001
7	30	0.1
8	30	0.01
9	30	0.001

4.3.8 Using window *n* to predict RMSE of window *n+1*

In a real-life scenario where a subsampling process is implemented, one will only have the info of the previous windows available, and those data may be subsampled. This means that the model should be able to predict the RMSE of the *n+1* window based on the features of the *n* window, which may be based on subsampled data.

To make this possible, the model must be trained on every combination of subsampling percentages in window *n* and *n+1*. One model is trained one all the different combinations. In a hypothetical scenario with 1000 windows and 20 sampling percentages the number of windows to train on would be 1000 x 20 x 20.

A new data frame needs to be created. It will be called **15Min-n+1-Dataframe**. In the data frame the target variable, RMSE, of the window *n+1* is shifted unto the features of window *n*. This target variable shifting process is done for all possible sampling percentages, X, for the window *n+1*. Window *n* now is represented 20 times in the new data frame, with 20 different RMSE values. The subsampling rate for the target variable will be represented with a feature called ”Subsampling Rate for Target Variable”. The subsampling rate for the features will be represented with a feature called ”Subsampling Rate for Features”. An overview of 15Min-n+1-Dataframe is provided in Figure 4.16.

The experiment is tested on the same three algorithms as in the baseline; a Linear regression, Random Forest Regression and Dummy Regression. The Linear Regression is trained out of the box with no hyperparameters tuned. The Random Forest Regression has the parameters n_estimators set to 10 and min_samples_leaf to 0.001. The Dummy Regression uses the mean of the target variable in the whole training set as predictor. The set selection is the one shown in Table 4.3 on page 46.

time	max	min	max-min	mean	std	std/mean	skew	RMSE	SR for Target Variable	SR for Features
2019-02-06 16:15:00+00:00	56.47	45.35	11.12	48.66	2.25	0.05	0.88	0.00	100	100
2019-02-06 16:15:00+00:00	56.47	45.35	11.12	48.75	2.34	0.05	0.77	2.39	60	15
2019-02-06 16:15:00+00:00	56.47	45.58	10.89	48.41	2.10	0.04	0.97	4.62	35	60
2019-02-06 16:15:00+00:00	56.47	45.47	11.00	48.63	2.58	0.05	0.98	15.63	15	20
2019-02-06 16:15:00+00:00	56.47	45.35	11.12	48.75	2.34	0.05	0.77	2.39	60	25
2019-02-06 16:15:00+00:00	56.47	45.47	11.00	48.63	2.58	0.05	0.98	15.63	15	95
2019-02-06 16:15:00+00:00	56.47	45.35	11.12	48.75	2.34	0.05	0.77	2.39	60	5
2019-02-06 16:15:00+00:00	56.47	45.47	11.00	48.63	2.58	0.05	0.98	15.63	15	40
2019-02-06 16:15:00+00:00	56.47	45.47	11.00	48.63	2.58	0.05	0.98	15.63	15	70
2019-02-06 16:15:00+00:00	56.47	45.35	11.12	48.75	2.34	0.05	0.77	2.39	60	100
2019-02-06 16:15:00+00:00	56.47	45.58	10.89	48.41	2.10	0.04	0.97	4.62	35	5
2019-02-06 16:15:00+00:00	56.47	45.47	11.00	48.63	2.58	0.05	0.98	15.63	15	45
2019-02-06 16:15:00+00:00	55.65	45.35	10.30	48.64	2.24	0.05	0.80	2.20	55	35
2019-02-06 16:15:00+00:00	56.47	45.47	11.00	48.63	2.58	0.05	0.98	15.63	15	50
2019-02-06 16:15:00+00:00	55.65	45.35	10.30	48.64	2.24	0.05	0.80	2.20	55	90
2019-02-06 16:15:00+00:00	56.47	45.47	11.00	48.63	2.58	0.05	0.98	15.63	15	65
2019-02-06 16:15:00+00:00	55.65	45.35	10.30	48.64	2.24	0.05	0.80	2.20	55	10
2019-02-06 16:15:00+00:00	56.47	45.47	11.00	48.63	2.58	0.05	0.98	15.63	15	55
2019-02-06 16:15:00+00:00	55.65	45.35	10.30	48.64	2.24	0.05	0.80	2.20	55	60
2019-02-06 16:15:00+00:00	56.47	45.47	11.00	48.63	2.58	0.05	0.98	15.63	15	60
2019-02-06 16:15:00+00:00	56.47	45.47	11.00	48.63	2.58	0.05	0.98	15.63	15	10
2019-02-06 16:15:00+00:00	55.65	45.35	10.30	48.64	2.24	0.05	0.80	2.20	55	55

Figure 4.16: *15min-n+1-Dataframe containing time-index and 22 instances of same 15-minute window from different shifted SubX-dataframes with accompanying features. "Subsampling Rate" shortened to "SR" for figure clarity. Rounding is performed for the same reason, but not done in the analysis.*

4.4 Method for performing an economic analysis

An imaginary installation with 22 sensors in a Wireless Sensor Network is used to do an economic analysis of different sampling strategies. The sensors should have a uniform sampling strategy so that the data produced is easier to analyze. As mentioned in chapter 2.3.2, the microcontroller is assumed to only have two different energy states. In sleep-mode, the energy-usage is 0. In on-mode, the energy-usage is the percentage of the subsampling. This mimics the process in the NTNU-paper, as mentioned in the aforementioned theory chapter. The percentage of the different sampling modes and their energy consumption can be seen in Table 4.5.

Table 4.5: *Sampling rates and their energy consumption.*

Sampling percentage	samples i in 15 min	Energy consumption
100 %	450	100 %
95 %	427	95 %
90 %	405	90 %
85 %	382	85 %
80 %	360	80 %
75 %	337	75 %
70 %	315	70 %
65 %	292	65 %
60 %	270	60 %
55 %	247	55 %
50 %	225	50 %
45 %	202	45 %
40 %	180	40 %
35 %	157	35 %
30 %	135	30 %
25 %	112	25 %
20 %	90	20 %
15 %	67	15 %
10 %	45	10 %
5 %	22	5 %

4.4.1 Pareto

After finding the models' optimal hyperparameters, the models are retrained on the whole training dataset, not split across K-folds. Should the models with the additional time-dependent features perform better, that is the dataset that will be chosen. If not, the models will be trained only on the time-independent features. The models are tested on the test dataset, which was set apart (week 16). This process is only done on the Using window *n* to predict RMSE of window *n+1*, as predictions on the same window are not of interest in a real-world context.

The data frame containing the results will have another feature added which represent the energy consumption The energy consumption is derived from Formula 2.5. The test accuracy is plotted against the energy consumption, and the Pareto front is found.

4.4.2 Cost-benefit analysis

In the cost-benefit analysis, the sensors are placed in an outdoor environment. Regardless of connectivity to the electricity grid or not, the sensors are the same. The sensors used in this analysis are not the Libelium-devices used in the data collection of the dataset. The sensors used and the information about them is provided by Soundsensing. A cost-benefit analysis is performed on different Senor Network alternatives. These are presented in Table 4.6.

Table 4.6: *Cost-benefit analysis - Alternatives.*

Alt.	Alternative description	Energy source	Sub-sampling rate
0	Grid Baseline	Connected to grid	100 %
1	Wireless baseline	Battery	100 %
2	NTNU SOA	Battery	7 %
3	Experiment	Battery	Pareto optimum

The sensors have a lifetime of 5-10 years, and the lifetime is set to 10 years to follow two cycles of the noise mapping demanded by [2]. The battery life will be dependent on the subsampling rate. A sensor that does continuous sampling is assumed to have a battery life of a month (30 days). The battery usage is assumed to be linear throughout the month. Thus, a reduction of energy consumption by 50 % would lead to a battery life of two months. A sensor using the NTNU SOA, which has a subsampling rate of 7 %, would have a battery life of 429 days, approximately 14 months.

$$Battery\ life_{days} = \frac{30\ days}{Subsampling\ rate} \tag{4.2}$$

where 30 is days before empty on full subsampling rate, and Subsampling rate is given in percentage.

Table 4.7: *Cost-benefit analysis - Battery information.*

Alt.	Energy source	Subsampling rate	Battery-life
Grid Baseline	Connected to grid	100 %	-
Wireless baseline	Battery	100 %	1 month
NTNU SOA	Battery	7 %	14 months
Experiment	Battery	Pareto optimum	-

The cost of each sensor is assumed to be 10 000 kr. This is not the real cost of the sensors that Soundsensing provides but is an example price used to make the cost-benefit analysis possible.

Every alternative is sending the data wirelessly over IoT-networks. The costs associated with this are a minimum of 20 kr per month. With continuous recording, one may assume a cost of 50 kr per month. The cost per extra percentage of subsampling is assumed to be linear, and thus 1 % of data sent costs 0.5 kr.

The installment of each sensor is associated with a cost. The alternative Grid Baseline has an extra cost in this regard, due to the need to connect to the electricity grid. The sensors will be placed outside, on a building's facade or other placements. An assumption is made that every installment has to be done by a certified electrician due to the cables which have to be laid out. The hourly price of an electrician is set to be 1000 kr [46]. Every installment is assumed to take 2 hours. For the sensors which use batteries as the energy source, the installment can be done by a person from the Soundsensing team and is assumed to take 1 hour. The associated cost of an hour's work is assumed to be 200 kr.

The batteries are rechargeable, and thus a new one does not have to be bought each time, but there is a cost with a person having to change the batteries. A battery swap is assumed to take 10 minutes, and the batteries are changed in the whole Wireless Sensor Networks as a whole. It is assumed 5 minute travel time between each sensor. The reason for using 5 minutes is the thought that the Wireless Sensor Network is in a specific local area. Thus, for 22 sensors, the associated time usage is five and a half hours. The battery swap can be done by a Soundsensing team member at the cost of 200 kr per hour.

$$Battery\ swap\ cost_{yearly} = \frac{12\ months}{Battery\ life} * 200\ kr \tag{4.3}$$

where 12 months is the number of months in a year, and Battery life is the number of months the battery lasts.

The cost of the electricity used by the sensor connected to the grid and the cost of the electricity used to recharge the batteries is assumed to be the same.

Table 4.8: *Cost-benefit analysis - Cost picture. (y) indicates the yearly cost.*

Alt.	Sensor	Installment	Data sending (y)	Battery swap (y)
Grid Baseline	10 000 kr	2000 kr	600 kr	0 kr
Wireless baseline	10 000 kr	200 kr	600 kr	2400 kr
NTNU SOA	10 000 kr	200 kr	240 kr	168 kr
Experiment	10 000 kr	200 kr	-	-

A discontinuation rate of 4 % is used for the Cost-benefit analysis, as decided in 2.4.2.

Chapter 5

Results

5.1 Prediction of RMSE

5.1.1 Using window *n* to predict RMSE of window *n*

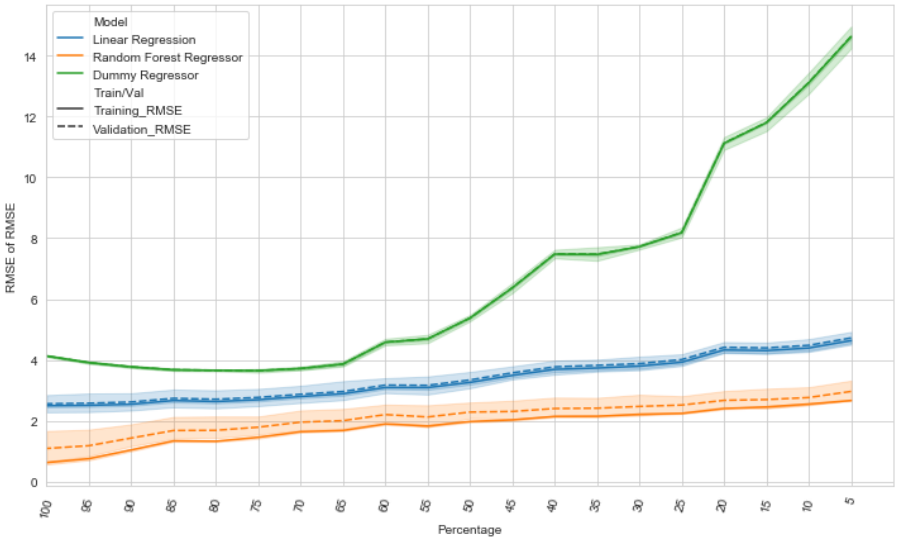


Figure 5.1: *Plot of model performance, Using window n to predict RMSE of window n. Accuracy (RMSE of RMSE) on the y-axis and Subsampling percentage on the x-axis.*

In Figure 5.1, one can see that both the Linear Regression and Random Forest Regression have a performance significantly better than the Dummy Regression for all subsampling percentages. This trend is reinforced for all subsampling percent-

ages below 65 %. The Random Forest Regression seems to be a little overfitted, which can be seen by the discrepancy between the training score and the validation score, also known as the model's variance. The Linear Regression has a lower variance than the Random Forest Regression but higher bias.

All three algorithms have an upward trend where the error is increasing with a lower subsampling rate. Both the Linear regression and the Random Forest Regression have a steady increase in error, indicating that the difference in error distribution is learned better by these models. The curve of the Dummy Regression has a much steeper slope, especially in the subsampling percentages below 25 %. Since the Dummy Regression predicts the same mean-value for all samples, this is to be expected.

5.1.2 Using window *n* to predict RMSE of window *n+1*

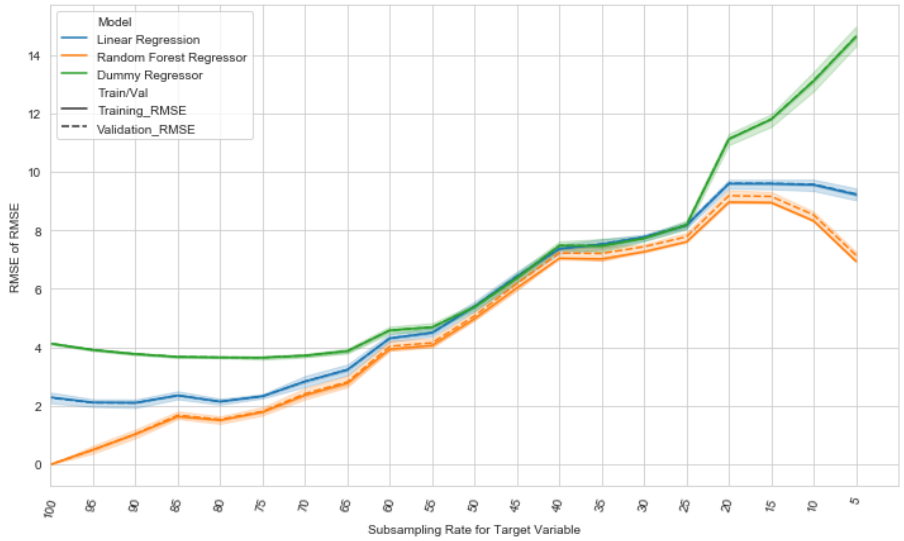


Figure 5.2: *Plot of model performance, Using window n to predict RMSE of window n+1. Accuracy (RMSE of RMSE) on the y-axis and Subsampling percentage on the x-axis.*

When looking at Figure 5.2, one can observe that it is possible to split the three models' performance into three parts. From 100 % to 60 %, from 60 % to 25 % and from 25 % to 5 %. From 60 % to 25 %, the models perform quite similar, while in the first and last part, the Random Forest Regression outperforms both the Linear Regression and the Dummy Regression.

The variance for all three algorithms is low, but the bias is markedly high, especially for low subsampling rates, with RMSE of RMSE up to 8 L_{Aeq} for subsampling rates around 25. The subsampling rate of 25 has another interesting char-

acteristic. Below this subsampling rate, both the Linear Regression and Random Forest Regression perform better than the Dummy Regression. This is particularly true for Random Forest Regression.

5.2 Prediction of RMSE with additional time-dependent features

5.2.1 Using window *n* to predict RMSE of window *n*

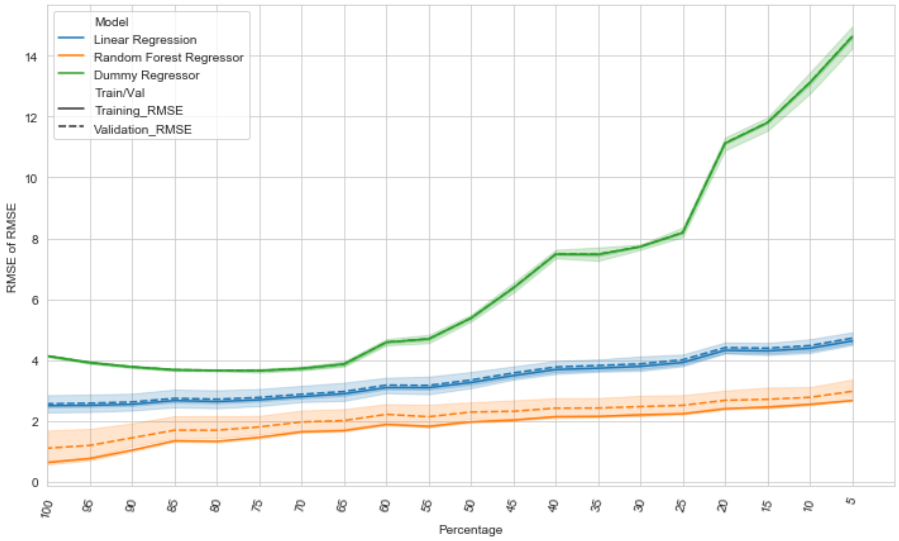


Figure 5.3: *Plot of model performance, Using window n to predict RMSE of window n with additional time-dependent features. Accuracy (RMSE of RMSE) on the y-axis and Subsampling percentage on the x-axis.*

Figure 5.3 shows no difference from Figure 5.1. This claim is verified when checking each model's performance against each other through a manual check of each score point in python.

5.2.2 Using window *n* to predict RMSE of window *n+1*

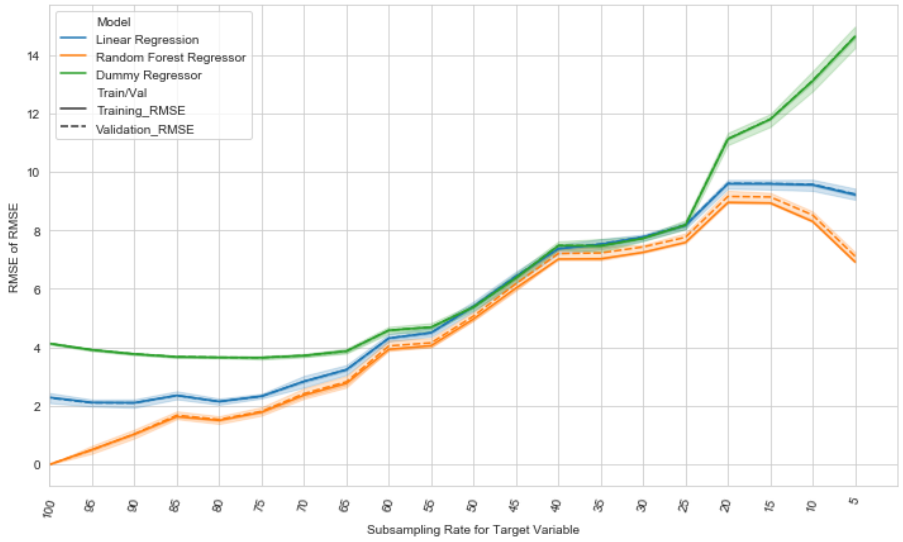


Figure 5.4: *Plot of model performance, Using window n to predict RMSE of window n+1 with additional time-dependent features. Accuracy (RMSE of RMSE) on the y-axis and Subsampling percentage on the x-axis.*

Figure 5.4 shows no difference from Figure 5.2. This claim is verified when checking each model's performance against each other through a manual check of each score point in python.

5.3 Economic impact of sampling strategies

5.3.1 Pareto optimization

To find the Pareto front, the models are re-trained on the full training dataset, as mentioned in 4.4.1. The re-training is done on the features which are not time-dependent since both 5.2.1 and 5.2.2 showed no difference in performance with the compared models.

The new re-trained models' performance and trends were checked against the previous models trained through the K-folds cross-validation. The model was determined to follow the same trends, and the Pareto plot was made, as presented in Figure 5.5.

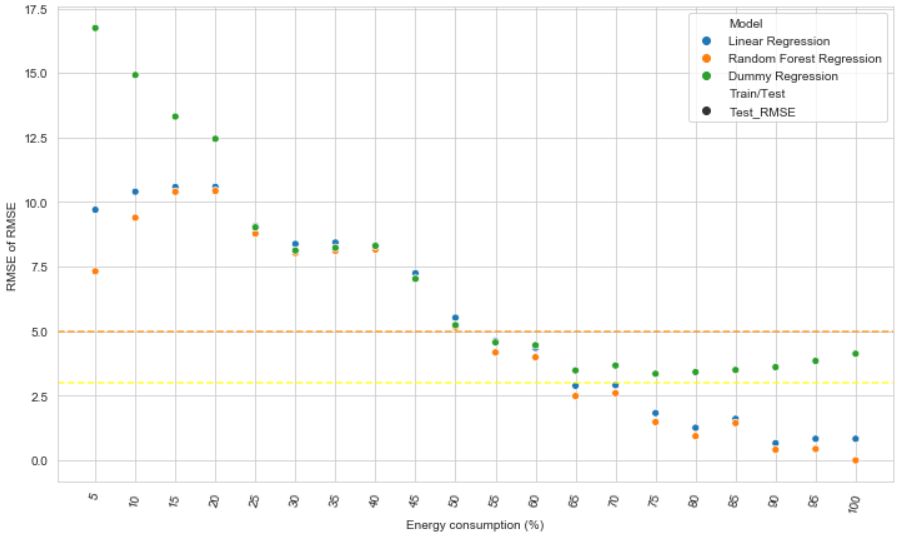


Figure 5.5: *Pareto plot. Using window n to predict RMSE of window n+1 without additional time-dependent features. Accuracy (RMSE of RMSE) on the y-axis while Energy consumption on the x-axis. The yellow dotted line (- - -) marks the threshold for error at ± 3 L_{Aeq}. The orange dotted line (- - -) marks ± 5 L_{Aeq}, which is deemed as an audible difference.*

The Pareto front will be the points where the optimal resource balance is. From Energy consumption 5 % to 25 %, the Random Forest Regression is creating the Pareto Front. From 25 % to 55 %, the three algorithms' Pareto points are almost indistinguishable. From energy consumption 100 % to 55 %, the Pareto front again lies at the points of the Random Forest Regression.

The threshold for error, which is marked with the yellow dotted line, is ± 3 L_{Aeq}. Thus the optimal point is where the energy consumption is the lowest, and the ac-

curacy (RMSE of RMSE) has not crossed the threshold for error. This point lies in the Pareto points of the Random Forest Regression and has an energy consumption of 65 % and an accuracy (RMSE of RMSE) of 2.5.

5.3.2 Cost-benefit analysis

The Pareto optimum of the experiment is used to do the Cost-benefit analysis. With a subsampling rate of 65 %, the battery life of a sensor is found through Formula 4.2. Thus, the Alternative Experiment has a battery life of 46 days. An overview of the battery information across all alternatives is shown in Table 5.1.

Table 5.1: *Cost-benefit analysis - Battery information - With results from the experiment.*

Alt.	Energy source	Subsampling rate	Battery-life
Grid Baseline	Connected to grid	100 %	-
Wireless baseline	Battery	100 %	1 month
NTNU SOA	Battery	7 %	14 months
Experiment	Battery	65 %	1.5 months

The yearly data sending cost is $0.5kr * 65 * 12 = 390kr$. The yearly battery swap cost with the Alternative Experiment is 1600 kr, as found through Formula 4.3. An overview of the cost information across all alternatives is shown in Table 5.1.

Table 5.2: *Cost-benefit analysis - Cost picture. (y) indicates the yearly cost - With results from the experiment.*

Alt.	Sensor	Installment	Data sending (y)	Battery swap (y)
Grid Baseline	10 000 kr	2000 kr	600 kr	0 kr
Wireless baseline	10 000 kr	200 kr	600 kr	2400 kr
NTNU SOA	10 000 kr	200 kr	240 kr	168 kr
Experiment	10 000 kr	200 kr	390 kr	1600 kr

The costs in Table 5.2 were multiplied by the number of sensors in the Wireless Sensor Network, 22, and a cost-benefit analysis was performed on the alternatives. The results are shown in Table 5.3

The alternative with the best net value is Alternative NTNU SOA, which uses batteries as an energy source and a subsampling rate of 7 %. The Net value is 326 656 kr. Following is the 0-alternative, the Grid Baseline, with a net value of 360 910 kr.

Table 5.3: *Results of Cost-benefit analysis*

Alt.	Energy source	Subsampling rate	Net value
Grid Baseline	Connected to grid	100 %	360 910 kr
Wireless baseline	Battery	100 %	789 162 kr
NTNU SOA	Battery	7 %	326 656 kr
Experiment	Battery	65 %	608 938 kr

With its subsampling rate of 65 % found through Pareto optimization, Alternative Experiment had a net value only better than the Wireless baseline. This is mostly due to the lower costs associated with the battery swaps.

Chapter 6

Discussion

6.1 Discussion of Materials and Methods

6.1.1 Data collection

There are multiple factors that may have affected the data collection and the resulting sensor data. One of these is the knowledge the student have of being monitored. This may have affected the students who were spending their time in Koopen and contributed to either lower and higher noise levels than normal. This factor is counterweighted by the length of the data collection period. Nine weeks would most likely have given the students time to adjust to the fact that their noise levels are being recorded. Koopen is often used for experiments, and this makes the students less likely to be affected by the data collection.

Another factor that may have affected the sound levels recorded is the distance from the most significant noise source, the students, to the Libelium recording devices. The Libelium devices were placed 2.5 meters above the students due to whiteboards hanging on the walls. Since the collected data is looked upon as the ground truth, the distance should not affect the prediction, but it may have altered the recorded soundscape.

6.1.2 Choice of master sensor

When choosing sensor node 03 as the master sensor and constructing the O.G Dataset, large amounts of data are discarded from the dataset. Even though some of the data from the other sensor nodes are used when imputing the weighted mean of the available sensors, the rest of the data from the sensor nodes is not used in the analysis. This is a simplification made due to the time and resource restraints of a master thesis. A more sophisticated analysis could involve all sensors in the

training process. Another possible route could have been to use the other sensors as different test sets, to see if the model is generalizable to other sensors. One would have to make sure that there was no data leakage, in that the time frame which is predicted in the other sensors does not overlap with any of the training or validation data. Neverteheless, even when taking measures like this, one would still have some data leakage due to the proximity of the sensors towards each other, and the placement in the same room. To create a dataset where the generalizability is more in focus, sensor nodes could be placed in different rooms and areas of the building.

6.1.3 Imputation process

The imputation process is also of interest. An imputation process was not done in the NTNU-paper, as it was stated that they discarded the windows with missing values. These results could not be reproduced by the author of this thesis.

In Figure 4.7, it can be observed that there are no windows with 0 missing measurements. The lowest amount of missing measurements found in a window was 6. This finding led to the decision to impute the missing measurements.

As seen in Figure 4.3, sensor node 05 overall has higher values for the measurements. When re-scaling the measurements by a constant, 0.9, an assumption is made that the device's calibration is wrong. Still, there is the possibility that the corner sensor node 05 is placed in is a particularly noisy corner, with ventilation or another constant noise in the very near vicinity, and that the data representation is correct. Because the dataset was acquired from external sources and that the Corona pandemic has made all universities close down their buildings, it was not easy to check if the assumption was correct or not, and the decision had to be made based only on the data. Thus, the decision to re-scale all values of sensor node 05. When re-scaling the L_{Aeq}-values, a change is made to the noisescape, and the noise environment has changed from what it was originally, something which may have affected the results.

Another challenge in the imputation process is applying the weighted mean across the sensor nodes for the imputation of the missing measurements in sensor node 03. This is a rather simple imputation and could have been strengthened if metadata regarding people's position in the room was available. With more time, a prediction model could have been created specifically for imputation, based on the data from the other sensors. Then the different imputing processes could have been compared to each other and the best used.

6.1.4 Choice of the time interval

The choice of the time interval is vital in the experimental setup. In the NTNU-master there were other $L_{Aeq,T}$ time intervals found to have better robustness,

specifically $L_{Aeq,8h}$ and $L_{Aeq,1h}$. However, by choosing 15-minute intervals as the time interval, it has the balance of being precise enough that a student may check for the interval each 15 minute and it is not too granular so that it has too much information for a student looking for a silent place to study. This is supported by the NTNU-paper, which used the same time interval.

6.1.5 Masking methods

The decision to choose the random masking method, MM2, instead of the masking method with the set interval, of masking frequency, MM1, is of considerable interest when looking at the results. Even though the flexibility of MM2 was determined to outweigh the more stable MM1, the decision was based on limited testing. If the one removed the demand for all the intervals to be exactly the same in MM1, one could have used this for all possible sampling percentages as well. This would have resulted in some of the windows being 2 seconds longer.

A different masking method may have given different answers, especially when looking at the subsampling rates below 20 %. This does not seem to be the case in either of the predictions in chapter 5.1.1 and chapter 5.1.2. Both the Linear Regression and Random Forest Regression did not have a significant worsening of performance in the area with subsampling rates below 20 %. In fact, in chapter 5.1.2, as seen in Figure 5.2, the Linear Regression has a flattening of performance, while the Random Forest Regression performs better in the aforementioned subsampling area.

MM2 cannot be called a true random masking method due to the fact that it is copied across all windows. If the random masking was performed once every window, then the masking method is more similar to a random selection. The frequency chosen for making the training data, and also the subsampling process in the evaluation and test data may have impacted the performance. The distribution of the elements picked can be seen in Figure 4.10. Some bins are a bit over-represented with elements compared to the neighboring bins, but it does not seem like a big problem. To counteract this uncertainty, a loop with ten different random seeds could have been implemented. Then the models could have been ran, and the performances checked against each other to see how much the masking method affected the results.

6.1.6 Set selection

To minimize the variance in the accuracy of the models, the choice was made to use K-folds cross-validation. The folds were split based on the week number and then paired through a random selection method. The different sets which were placed together, as seen in Table 4.3, may have affected the performance. To further increase the evaluation process's trustworthiness, the paired weeks in the validation

sets could have been redone another time to test if different pairs might affect the performance.

This K-folds cross-validation uses an independent random selection of weeks to select the folds. For time-series, this is not a good practice. In the first fold in Table 4.3, one can see that week 11 is in the training set while weeks 10 and 12 are in the validation set. Due to the expectation that these weeks will be quite similar, this results in a violation of independence and data leakage. Another factor that weakens the choice of using independent random selection to select the folds is when thinking about a real-world prediction. If the current week is week 10, future measurements are not available, and there are no future weeks.

6.1.7 Feature extraction

The decision to only use the column with L_{Aeq}-values was taken due to the goal to test the models with features that were time-independent. Later, a decision was made to extend the search to also look at some columns based on the time-column as well. The two chosen features were the two most important features in the predictions of the noise predictors in the NTNU-paper, day of week and hour. In Figure 4.4, the differences in L_Aeq values over both days and hours can be observed. There is a much bigger presence of noise on the weekdays, and specifically in the timeframe of 06:00 till 19:00.

Other features that were used in the NTNU-paper and that could also have been included are holiday, schedule and quarter. Schedule here would describe if there were any planned events in Koopen. Based on Figure 4.4, another feature of interest could have been a night/morning/midday/afternoon feature.

6.1.8 Algorithm selection

There are several considerations that need to be evaluated. One of these is the choice of the baseline. The Dummy Regression predicts based on the mean. When looking at Figure 4.3, Figure 4.9 and Table 4.1, one can observe that although the measurements are mostly in the area between 40 and 50 L_{Aeq}, an algorithm that always predicts the mean will be limited. A baseline with more flexibility, such as an algorithm that is randomly predicted in the interval between 40 and 50 L_{Aeq} might have provided more insight.

The number of other algorithms tested could also have been increased. There are several other algorithms that could have been of interest. This is especially true when looking at the Linear Regression. Here, either Lasso Regression or Ridge Regression could have worked with collinear features, as mentioned in chapter 2.2.4.

6.1.9 Learning strategy - offline and online

The learning strategy for the experimental setup was based on an offline-learning strategy, as mentioned in Chapter 2.3.1. This was a good starting point since it gives an overview of what kind of performance is possible with full historical data. The long training period made the strategy not as adaptable to changes, and it vulnerable to changes that happen after the training period. This was tried remedied with K-fold cross-validation.

An online learning strategy would have been less susceptible to the vulnerabilities of the offline learning strategy. Here one could have used e.g., only an estimate from the previous week to estimate the future values. Different problems do occur instead, most importantly, the question "Does the features available contain enough information that the performance of the model will be good enough?". The results in this master thesis indicate that no, they do not.

6.2 Model performance

6.2.1 Using window *n* to predict RMSE of window *n*

The Random Forest Regression performs the best of the algorithms tested, as seen in Figure 5.1. It is a little overtrained, but this has been remedied with the min_samples_leaf set to 0.001, as found through the small grid-search mentioned in chapter 4.3.8. The steady increase of both the Linear Regression and Random Forest Regression show that they are able to predict the error even for subsampling rates lower than 25 %, which have a much different distribution of errors, as observed in Figure 4.3.2. This is also the area where the Dummy Regression has the most substantial increase in performance error and shows that the Dummy Regression, which uses the mean as method for prediction, struggles to predict correctly when the error distribution widens. The Dummy Regression never gets under the threshold of error at ± 3 L_{Aeq}, something which both the Linear Regression and the Random Forest Regression manages to. The Random Forest Regression has performance better than the threshold of error for all subsampling rates, a positive sign for the baseline of the experiment.

6.2.2 Using window *n* to predict RMSE of window *n+1*

In a real-life scenario where a subsampling process is implemented, one will only have the info of the previous windows available, and those data may be subsampled. Figure 5.2 shows vastly different performance scores for both the Linear Regression and the Random Forest Regression than in the baseline presented over. The Dummy Regression performs the same, which is natural given that it uses the mean of the whole training set as the predictor.

One can split the three models' performance into three parts. From 100 % to 60 %, from 60 % to 25 % and from 25 % to 5 %. From 60 % to 25 %, the models perform quite similar, while in the first part and the last part, the Random Forest Regression outperforms both the Linear Regression and the Dummy Regression.

In the middle part, from 60 % to 25 %, it seems that neither the Linear Regression nor the Random Forest Regression manages to beat predicting with the mean of the previous window. Similarly, it is when the two models reach this subsampling rate that they exceed the error of threshold.

6.2.3 Prediction of RMSE with additional time-dependent features

The predictions on both Using window *n* to predict RMSE of window *n* and Using window *n* to predict RMSE of window *n+1* show no difference when the additional time-dependent features are added. Figure 5.3 shows little difference from Figure 5.1. The same can be seen in Figure 5.4, which shows little difference from Figure 5.2.

The fact that the models did not perform better with additional features derived from the time feature is surprising. The NTNU-paper showed that the two extra features chosen to be added into the model were the most important for predicting their noise predictors. This fact adds uncertainty into the performance of this thesis' models and asks the question, is the metadata derived from the noise level, the L_{Aeq}, that much more important than the time. When looking at Figure 4.4 there are definite trends that are dependent on the time, both the day and the hour. Thus one would expect that adding these two features would increase the performance of the models.

6.2.4 Model selection and optimization

A possibility to find a better model could have been to implement a more extensive grid-search in search of the best model. One could also have used other models, as mentioned earlier in discussion.

When comparing the models in this thesis with the subsampling percentage found to be optimal in the NTNU-paper, it is the model sing window *n* to predict RMSE of window *n* that is of interest. This is due to the setup of the experiment for the models Using window *n* to predict RMSE of window *n+1*, which utilized subsampled 15 minute windows as input, something which the NTNU-paper does not. When having that in mind, one can observe in Figure 5.1 that the difference in performanceis not that big between the NTNU SOA and the Random Forest Regression in this experiment. This was not taken into consideration when performing the economic analysis.

6.3 Economic impact of sampling strategies

A significant weakness of the Pareto plot is Formula 2.5, which is used to calculate energy consumption. Even though the same process was used in the NTNU-paper, it is a big simplification of the different energy states that a microcontroller has. Thus the energy consumption and the sampling rate have a 1:1 ratio, which probably does not generalize to the real world. Both the Linear Regression and the Random Forest Regression perform better than the error of audible threshold down to a sampling rate of 65 %, which indicates that a 35 % energy reduction can be gained with the presumptions made. This performance would have been changed if a secondary energy source such as solar panels had been introduced to the equation. Other factors that may affect energy consumption are the energy usage in the idle state, to keep the RAM running and the energy usage of the modem, which sends the data.

In the Cost-benefit analysis, the Pareto optimum found through the experiment is used. The implication of using the findings on a dataset on an indoor environment to an outdoor space implies that the conclusions done need to be treated with care. Indoor noisescapes have different characteristics than outdoor noisescapes. However, the limited access to a high-quality dataset of continuous noise measurements over a long period of time means that the indoor environment dataset was used. TThe Alternative NTNU SOA, which was ultimately found to have the best net value, was also derived from the same indoor dataset used in this master thesis. The NTNU SOA was found to have approximately a 35 000 kr better net value than Alternative Grid Baseline. Since the Cost-benefit analysis only involves economic criteria, this gap could have changed if different criteria were introduced to the analysis. This is particularly true in the realm of facade interventions, but also flexibility in where the sensors may be placed. In Alternative Grid Baseline, the sensors need to have access to the electrical grid, limiting the possible placements. This is not a factor in the alternatives where batteries are the energy source.

The costs which the Cost-benefit analysis was based on are using several assumptions. If some of these assumptions were replaced by facts, the uncertainty in the analysis's uncertainty would have been lessened. The highest variable cost to wireless sensor networks focusing on noise estimations is the cost of manual labor when changing batteries. This is why a subsampling strategy will have such a big impact on the viability of different business cases. The cost of installment for Alternative Grid Baseline could be reduced if a framework agreement was made with an electrician firm.

6.4 Implications for environmental noise monitoring

A lowering of cost means that applications which earlier were not economically feasible, now are. With real-time feedback on the noise levels in a city, the munici-

pality has much more information available. Decisions made are more data-driven and less based on human intuition. The noise maps demanded by the EU will also be easier to create due to the more stable data sources and may achieve better accuracy than with the usage of complicated simulation models.

This is not necessarily only relevant for municipalities, but businesses may also have substantial benefits from having an understanding of the noise levels in their working environment. Noise has been shown to impact the concentration and workflow, and a better understanding of the noisescape of the workplace may lead to higher productivity.

Chapter 7

Conclusions & Further work

A dataset was procured from NTNU containing noise measurements over an 11 weeks. The data was preprocessed and used to train three models, a Linear Regression, a Random Forest Regression and a Dummy Regression. These were evaluated based on the accuracy (RMSE of RMSE). The goal was to find the subsampling rate, which both upheld the threshold of error and had the lowest possible energy consumption. The threshold for error was ± 3 L_{Aeq}, and the Pareto optimal point for the experimental setup was created by the Random Forest Regression model. It had an energy consumption of 65 % and an accuracy (RMSE of RMSE) of 2.5. A cost-benefit analysis was done on four different Wireless Sensor Network alternatives consisting of 22 sensors with different sampling strategies implemented. The alternative with the best net value was Alternative NTNU SOA, which used batteries as an energy source, a subsampling rate of 7 % and achieved a RMSE of ± 2 L_{Aeq} in their study. The Net value was 326 656 kr. The next best alternative was the 0-alternative, the Grid Baseline, with a net value of 360 910 kr. Alternative Experiment, with it's subsampling rate of 65 % found through Pareto optimization, had a net value of 608 938 kr, which was only better than the Wireless baseline of 789 162 kr.

If more time was available, an uncertainty analysis could have been performed in conjunction with the Cost-benefit analysis. Here the economic benefits could have been combined with benefits like e.g., the technical solution and innovations created through that and the intervention on facade and architecture. An expert group with Oslo municipality and Soundsensing team members could have been created to quantify the scores of the different alternatives.

Another part of the further work is creating a prediction model specifically for the imputation of missing values based on meta-data from the other sensors. This may have lead to more correct values and, subsequently, better performing models. The models could also have been improved if more time-based features were included,

even though the tests done showed that the time-based features used had no impact on the model's performance.

Another way of handling the problem could have been to use reinforcement learning instead of supervised learning. A big challenge in the supervised learning method was to find a suitable target variable. This challenge could have been avoided by using reinforcement learning, and finding the optimum for the multiple objectives (accuracy and energy usage) may have been easier.

Bibliography

[1] W. H. O. et al., "Burden of disease from environmental noise: Quantification of healthy life years lost in europe," *Quantification of healthy life years lost in Europe*, p. 126–126, 2011.

[2] E. Directive, "Directive 2002/49/ec of the european parliament and the council of 25 june 2002 relating to the assessment and management of environmental noise," *Official Journal of the European Communities*, vol. 189, 2002.

[3] P. Mioduszewski, J. Ejsmont, J. Grabowski, and D. Karpiński, "Noise map validation by continuous noise monitoring," *Applied Acoustics - APPL ACOUST*, vol. 72, pp. 582–589, 07 2011.

[4] J. Mayes, "Urban noise levels are high enough to damage auditory sensorineural health," *Cities & Health*, pp. 1–7, 02 2019.

[5] J. P. Bello, C. Silva, O. Nov, R. L. Dubois, A. Arora, J. Salamon, C. Mydlarz, and H. Doraiswamy, "Sonyc: A system for monitoring, analyzing, and mitigating urban noise pollution," *Commun. ACM*, vol. 62, no. 2, p. 68–77, Jan. 2019. [Online]. Available: https://doi.org/10.1145/3224204

[6] B. Vinci, A. Tonacci, C. Caudai, P. Rosa, L. Nencini, and L. Pratali, "The senseable pisa project: Citizen-participation in monitoring acoustic climate of mediterranean city centres," *CLEAN - Soil Air Water*, vol. 45, 05 2016.

[7] X. Sevillano, J. C. Carrié, F. Alías, P. Bellucci, L. Peruzzi, S. Radaelli, P. Coppi, L. Nencini, A. Cerniglia, A. Bisceglie, R. Benocci, and G. Zambon, "Dynamap – development of low cost sensors networks for real time noise mapping," *Noise Mapping*, vol. 3, 01 2016.

[8] J. Picaut, A. Can, J. Ardouin, P. Crépeaux, T. Dhorne, D. Écotière, M. Lagrange, C. Lavandier, V. Mallet, C. Mietlicki, and M. Paboeuf, "Characterization of urban sound environments using a comprehensive approach combining open data, measurements, and modeling," *The Journal of the Acoustical Society of America*, vol. 141, no. 5, pp. 3808–3808, 2017. [Online]. Available: https://doi.org/10.1121/1.4988416

[9] V. M. Erichsen, "Soundsensing landet avtale med justisdepartementet: Skal måle støy fra nytt beredskapssenter," *Shifter*. [Online]. Available: https://shifter.no/nyheter/soundsensing-landet-avtale-med-justisdepartementet-skal-male-stoy-fra-nytt-beredskapssenter/106599

[10] "Støy," https://www.vegvesen.no/_attachment/735648/binary/1002701?fast_title=St\OT1\oyark.pdf, accessed: 2020-06-28.

[11] in *Vehicle Noise and Vibration Refinement*, X. Wang, Ed. Woodhead Publishing, 2010, p. xiii.

[12] I. M. V. Bosch, "Efficient noise measurement with energy constrained iot nodes - a case study on working environment quality," Master's thesis, NTNU, XXX, 2019.

[13] F. A. Kraemer, F. Alawad, and I. M. V. Bosch, "Energy-accuracy tradeoff for efficient noise monitoring and prediction in working environments," in *Proceedings of the 9th International Conference on the Internet of Things*. New York, NY, USA: Association for Computing Machinery, 2019. [Online]. Available: https://doi.org/10.1145/3365871.3365885

[14] E. H. Berger, *The Noise Manual*. AIHA, 2003.

[15] S. Gelfand, *Essentials of Audiology*, ser. Thieme Publishers Series. Thieme, 2009. [Online]. Available: https://books.google.no/books?id=_tcPcPTwNQoC

[16] "Lesson: Ultrasonic devices at the speed of sound!" https://www.teachengineering.org/lessons/view/mis-2227-ultrasonics-uses-arduino-ultrasound-technology, accessed: 2020-03-27.

[17] T. P. Classroom, "Sound waves and music - lesson 2 - sound properties and their perception," February 2015.

[18] B. Shield and J. Dockrell, "The effects of noise on children at school: A review," *Building Acoustics*, vol. 10, pp. 97–116, 06 2003.

[19] "Electroacoustics – Sound level meters – Part 1: Specifications," International Electrotechnical Commission, Geneva, CH, Standard, 2013.

[20] N. B. Stremmel and C. J. Struck, "Acoustical standards news," *The Journal of the Acoustical Society of America*, vol. 142, no. 2, pp. 631–640, 2017. [Online]. Available: https://doi.org/10.1121/1.4996107

[21] L. Brocolini, C. Lavandier, M. Quoy, and C. Ribeiro, "Measurements of acoustic environments for urban soundscapes: Choice of homogeneous periods, optimization of durations, and selection of indicators," *The Journal of the Acoustical Society of America*, vol. 134, pp. 813–21, 07 2013.

[22] Miljødirektoratet, "Veileder m-290 (2015). måling av støy fra industri. immisjonsmålemetode." \OT1\textquotedblrighthttp://www.miljodirektoratet.no/Documents/publikasjoner/M290/M290.pdf, accessed: 2020-06-22.

[23] I. P. Mary, "Imputing the missing values in iot using estcp model," *International Journal of Advanced Research in Computer Science*, vol. 8, pp. 532–536, 09 2017.

[24] S. Raschka and V. Mirjalili, *Python machine learning*. Packt Publishing Ltd, 2017.

[25] T. C. Nowe, "Detection and quantification of rot in harvested trees using convolutional neural networks," Master's thesis, NMBU, http://hdl.handle.net/11250/2620961, 2019.

[26] D. Schreiber-Gregory, "Regulation techniques for multicollinearity: Lasso, ridge, and elastic nets," 01 2018.

[27] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001. [Online]. Available: http://dx.doi.org/10.1023/A%3A1010933404324

[28] "sklearn.dummy.dummyregressor," https://scikit-learn.org/stable/modules/generated/sklearn.dummy.DummyRegressor.html, accessed: 2020-00-21.

[29] "Microcontrollers will regain growth after 2019 slump," https://www.icinsights.com/news/bulletins/Microcontrollers-Will-Regain-Growth-After-2019-Slump/, accessed: 2020-04-14.

[30] R. Steuer, *Multiple Criteria Optimization: Theory, Computation, and Application*, ser. (WILEY SERIES IN PROBABILITY AND MATHEMATICAL STATISTICS). Wiley, 1986. [Online]. Available: https://books.google.no/books?id=0H9jQgAACAAJ

[31] M. Bjorkli, "Resource-aware acoustic event classification," Master's thesis, NTNU, XXX, 2019.

[32] S. Cellini and J. Kee, *Cost-Effectiveness and Cost-Benefit Analysis*, 10 2015, pp. 636–672.

[33] P. for Finansdepartementet, "Nou 2015: 1. produktivitet – grunnlag for vekst og velferd — produktivitetskommisjonens første rapport," *Noregs offentlege utgreiingar*, vol. 1, 2015.

[34] R. from a committee appointed by Royal Decree of 18 February 2011, "Nou 2012: 16. cost-benefit analysis," *Noregs offentlege utgreiingar*, vol. 12, 2012.

[35] B. A. Kitchenham, "Procedures for performing systematic reviews," 2004.

[36] "Studentgründere utvikler nytt sensorsystem for støy," https://www.nmbu.no/aktuelt/node/37513, accessed: 2020-04-10.

[37] *ACM Trans. Auton. Adapt. Syst.*, vol. 12, no. 1, 2017.

[38] W. Cheng, S. M. Erfani, R. Zhang, and R. Kotagiri, "Learning datum-wise sampling frequency for energy-efficient human activity recognition," in *AAAI*, 2018.

[39] D. Trihinas and G. Pallis, "Adam: An adaptive monitoring framework for sampling and filtering on iot devices," 10 2015.

[40] T. E. Oliphant, *A guide to NumPy*. Trelgol Publishing USA, 2006, vol. 1.

[41] W. McKinney *et al.*, "Data structures for statistical computing in python," in *Proceedings of the 9th Python in Science Conference*, vol. 445. Austin, TX, 2010, pp. 51–56.

[42] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[43] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing in science & engineering*, vol. 9, no. 3, pp. 90–95, 2007.

[44] M. Waskom, O. Botvinnik, D. O'Kane, P. Hobson, S. Lukauskas, D. C. Gemperline, T. Augspurger, Y. Halchenko, J. B. Cole, J. Warmenhoven, J. de Ruiter, C. Pye, S. Hoyer, J. Vanderplas, S. Villalba, G. Kunter, E. Quintero, P. Bachant, M. Martin, K. Meyer, A. Miles, Y. Ram, T. Yarkoni, M. L. Williams, C. Evans, C. Fitzgerald, Brian, C. Fonnesbeck, A. Lee, and A. Qalieh, "mwaskom/seaborn: v0.8.1 (september 2017)," Sep. 2017. [Online]. Available: https://doi.org/10.5281/zenodo.883859

[45] "Koopen - elektrobygget," https://www.ntnu.no/laeringsarealer/koopen/, accessed: 2020-04-29.

[46] "Prisliste oslo elektriske as fra 1.juli 2018," https://www.oslo-elektriske.no/sfiles/17/42/91/6/file/prisliste_oslo_elektriske_as_05_07_2018.pdf, accessed: 2020-06-27.

Appendix

Appendix A: Cost-benefit analysis

KVU Brønnøysundregistrene; Økonomiske forutsetninger					
Parameter	Min	Sanns.	Maks	Verdi	Kommentar
Diskonteringsfaktor				4,0 %	

Økk. parametre | Alt. 0 | Alt. 1 Wireless baseline | Alt. 2 NTNU SOA | Alt. 3 - Pareto optimum | Empty | Empty_2 | Empty_3

Figure 7.1: *The process of cost-benefit analysis. Discontinuation rate.*

0-alternative - Grid Baseline				
Simulerte kostnader	**Min.**	**Samns.**	**Maks.**	**Simulert**
Investeringskostnader	12000	12000	12000	12000
Data-sending	600	600	600	600
Battery swap				
SUM kostnader				**12600**

Investeringer, periodisering	**Verdi**	**2020**	**2021**	**2022**	##	##	##	##	##	##	**2029**	**2030**	###	###	###	**2051**
Investeringskostnader	12 000	100 %														

Kapitalstrømmer, Alternativ 0	**2020**	**2021**	**2022**	##	##	##	##	##	##	**2029**	**2030**	###	###	###	**2051**
Investeringskostnader	12000														
Data sending	600	600	600	##	##	##	##	##	##	600					
Battery swap															
SUM kostnader	**12600**	**600**	**600**	##	##	##	##	##	##	**600**					

Nåverdi 0-alternativet	**16 405**

Figure 7.2: The process of cost-benefit analysis. Alternative 0.

Alternative 1: Wireless baseline

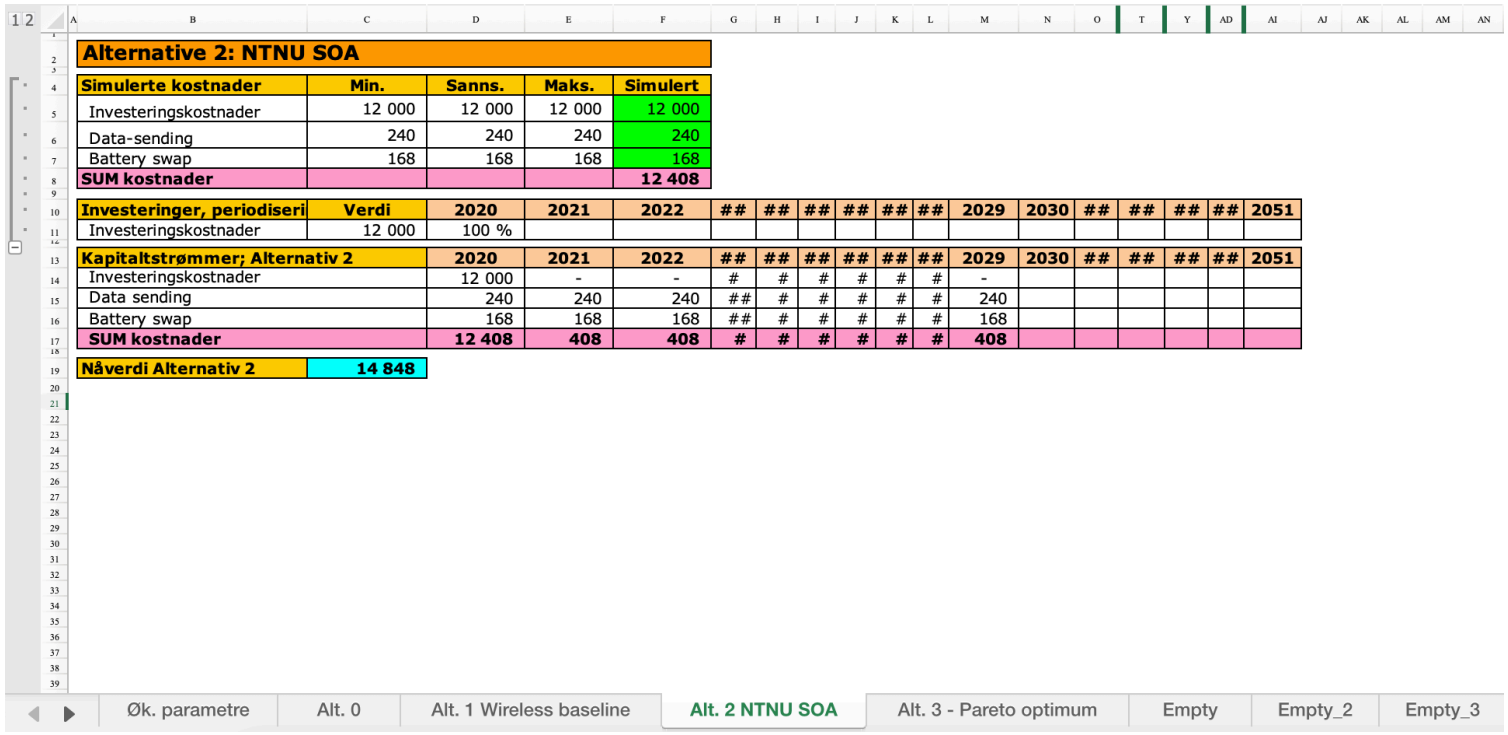
Simulerte kostnader	Min.	Sanns.	Maks.	Simulert
Investeringskostnader	12000	12000	12000	12000
Data-sending	600	600	600	600
Battery swap	2400	2400	2400	2400
SUM kostnader				**15000**

Investeringer, periodiser	Verdi	2020	2021	2022	##	##	#	#	##	##	2029	2030	##	##	##	2051
Investeringskostnader	12 000	100 %														

Kapitalstrømmer, Alternativ 1	2020	2021	2022	##	##	#	#	##	##	2029	2030	##	##	##	2051
Investeringskostnader	12000														
Data sending	600	600	600	##	##	##	##	##	##	600					
Battery swap	2400	2400	2400	##	##	##	##	##	##	2400					
SUM kostnader	**15000**	**3000**	**3000**	**##**	**##**	**#**	**#**	**##**	**##**	**3000**					

Nåverdi Alternativ 1	35 871

Figure 7.3: The process of cost-benefit analysis. Alternative 1.



Alternative 2: NTNU SOA

Simulerte kostnader	Min.	Sanns.	Maks.	Simulert
Investeringskostnader	12 000	12 000	12 000	12 000
Data-sending	240	240	240	240
Battery swap	168	168	168	168
SUM kostnader				**12 408**

Investeringer, periodiseri	Verdi	2020	2021	2022	##	##	##	##	##	##	2029	2030	##	##	##	##	2051
Investeringskostnader	12 000	100 %															

Kapitaltstrømmer; Alternativ 2	2020	2021	2022	##	##	##	##	##	##	2029	2030	##	##	##	##	2051
Investeringskostnader	12 000	-	-	#	#	#	#	#	#	-						
Data sending	240	240	240	##	#	#	#	#	#	240						
Battery swap	168	168	168	##	#	#	#	#	#	168						
SUM kostnader	**12 408**	**408**	**408**	**#**	**#**	**#**	**#**	**#**	**#**	**408**						

Nåverdi Alternativ 2	14 848

Øk. parametre | Alt. 0 | Alt. 1 Wireless baseline | Alt. 2 NTNU SOA | Alt. 3 - Pareto optimum | Empty | Empty_2 | Empty_3

Figure 7.4: The process of cost-benefit analysis. Alternative 2.

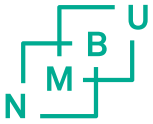
Alternativ 3: Pareto optimum				
Simulerte kostnader	**Min.**	**Sanns.**	**Maks.**	**Simulert**
Investeringskostnader	12 000	12 000	12 000	12 000
Data-sending	390	390	390	390
Battery swap	1 600	1 600	1 600	1 600
SUM kostnader				**13 990**

Investeringer, periodiserir	**Verdi**	**2020**	**2021**	**2022**	**##**	**2024**	**##**	**##**	**##**	**2029**	**2030**	**##**	**##**	**##**	**2051**
Investeringskostnader	12 000	100 %													

Kapitalstrømmer; Alternativ 3	**2020**	**2021**	**2022**	**##**	**2024**	**##**	**##**	**##**	**2029**	**2030**	**##**	**##**	**##**	**2051**
Investeringskostnader	12 000	-	-	#	-	#	#	#	-					
Data sending	390	390	390	##	##	##	##	##	390					
Battery swap	1 600	1 600	1 600	##	##	##	##	##	1 600					
SUM kostnader	**13 990**	**1 990**	**1 990**	**#**	**##**	**#**	**#**	**#**	**1 990**					

Nåverdi Alternativ 3	**27 679**

Figure 7.5: The process of cost-benefit analysis. Alternative 3.



Norges miljø- og biovitenskapelige universitet
Noregs miljø- og biovitskapelege universitet
Norwegian University of Life Sciences

Postboks 5003
NO-1432 Ås
Norway