NTNU

Norwegian University of
Science and Technology

Detection of Bowel Sounds

Kaja Kvello

Master of Science in Cybernetics and Robotics
Submission date: June 2018
Supervisor: Øyvind Stavdahl, ITK
Co-supervisor: Konstanze Kölle, ITK
Anders Lyngvi Fougner, ITK

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Abstract

This thesis describes the development and implementation of a bowel sound detection algorithm, which ultimately might be used as part of a meal detection system in an artificial pancreas. An artificial pancreas is an advanced blood glucose regulation system for patients with diabetes. In order to correctly provide insulin doses to the user, artificial pancreases need to detect when the patient is eating. One possible approach for meal detection is monitoring bowel sound occurrence frequency. This further necessitates a bowel sound detection system, such as the one described in this thesis.

The detection algorithm will consist of a combination of signal processing techniques for feature extraction, and pattern recognition in order to classify sound segments as either bowel sound segments or non-bowel sound segments. A linear support vector machine was chosen for classification in the final solution. The Python implementation of the detection system yielded promising results on a provided data set of sound measurements from the stomach. However, certain improvements and refinements should be done to the system before potential integration into an artificial pancreas. The thesis also briefly discusses these further development needs.

Chapter 1 explains the motivation behind the assignment, as well as the problem description and approach. Chapter 2 provides the most important theoretical background for the work described in this thesis. Chapters 3 and 4 describe the method and work towards the final product. These chapters also contain intermediate results which constitute the basis for decisions made throughout the process. Results obtained from the final solution can be found in chapter 5, while chapter 6 summarizes and concludes the report.

Sammendrag

Denne oppgaven beskriver utvikling og implementasjon av en deteksjonsalgoritme for tarmlyder, som muligens kan brukes som en del av et måltiddeteksjonssystem i en kunstig bukspyttkjertel. En kunstig bukspyttkjertel er et avansert blodsukkerreguleringssystem for pasienter med diabetes. For å korrekt tilføre insulindoser til brukeren, må kunstige bukspyttkjertler oppdage når pasienten spiser. En mulig metode for måltidsdeteksjon er å overvåke forekomstfrekvensen av tarmlyder. Dette krever videre et deteksjonssystem for tarmlyder, slik som beskrevet i denne avhandlingen.

Deteksjonsalgoritmen vil bestå av en kombinasjon av signalbehandlingsteknikker for egenskapsuttrekking, samt mønstergjenkjenning for å klassifisere lydsegmenter som enten tarmlydsegmenter eller ikke-tarmlydsegmenter. En lineær støttervektormaskin (*support vector machine*) ble valgt for klassifisering i den endelige løsningen. Python-implementasjonen av deteksjonssystemet ga lovende resultater på et gitt datasett med lydmålinger fra magen. Allikevel er det nødvendig med en viss grad av raffinering og forbedring av systemet før eventuell integrering i en kunstig bukspyttkjertel. Rapporten drøfter også disse videre utviklingsbehovene.

Kapittel **??** forklarer motivasjonen bak oppgaven, samt oppgavebeskrivelsen og tilnrmingen til problemet. Kapittel 2 gir den viktigste teoretiske bakgrunnen for arbeidet som er beskrevet i denne oppgaven. Kapitlene 3 og 4 beskriver metodene brukt i arbeidet mot sluttproduktet. Disse kapitlene inneholder ogs mellomliggende resultater som danner grunnlag for beslutninger som er gjort gjennom hele prosessen. Resultatene fra den endelige lsningen finner du i kapittel 5, mens kapittel 6 oppsummerer og konkluderer rapporten.

Preface

This master thesis is submitted as a part of the requirements for the M.Sc. degree within the field of Engineering Cybernetics at the Norwegian University of Science and Technology (NTNU). The assignment was initiated by the Artificial Pancreas Trondheim (APT) research group, and the thesis was written for the Department of Engineering Cybernetics at NTNU. This thesis is a derivative of the ongoing Ph. D work of Konstanze Kölle, and was carried out during spring 2017. The aforementioned Ph. D thesis is concerned with meal detection in artificial pancreas systems, and revealed the potential benefit of a bowel sound detection system as a part of the total meal detection solution.

The APT group provided the data set of sound measurements as a .wav-file, and provided the problem description, which asked for detection of the bowel sounds in the data set. Pattern recognition was mentioned as a possible method, but the final problem approach presented in section 1.3 was my own idea. My co-supervisor, Konstanze Kölle, provided relevant background literature in the initial stages of the work. Throughout the process she has also contributed with thoughts and discussion on my various ideas and suggestions for problem specification, solution methods and report structure. My main supervisor, Øyvind Stavdahl also came with brief input on the form of the report and with initial suggestions regarding signal processing methods. The Department of Engineering Cybernetics provided a work station and computer, and APT conveyed a wish of implementation in Python. Apart from this, no assistance was given in the fulfillment of this thesis. To not exploit the resources in the research group, no experts were hired to label the bowel sounds in the data set. All work of manual data marking was done by me, as explained in section 4.4.

I would like to thank Konstanze Kölle for all her guidance throughout this semester. Also Øyvind Stavdahl for his comments and help from across the globe. Further, I want to thank Anja Bøe for sharing her medical knowledge of bowel sounds and Vincent Simensen, both for help with proofreading and for comfort in my times of discouragement. Lastly, thanks to my parents for their unwavering support throughout my student years.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Abbreviations

APT	=	Artificial Pancreas Trondheim
BG	=	Background
BS	=	Bowel sound
CGM	=	Continuous glucose monitoring
MNLE	=	Mean nonlinear energy
PSD	=	Power spectral density
SGD	=	Stochastic gradient descent
STFT	=	Short-time Fourier transform
SVM	=	Support vector machine

CHAPTER

1

INTRODUCTION

1.1 Motivation

1.1.1 Diabetes

Digestion and cell respiration are some of the most important functionalities in the human body. In normal digestion the carbohydrates found in food are converted into glucose, which is then transported into the bloodstream. The increased glucose level in the blood triggers the production of insulin, a hormone which notifies the cells of the body to absorb glucose from the blood. This glucose is then used in the cell's production of energy. [1]

According to [2]: "Diabetes is a chronic disease that occurs when the pancreas is no longer able to make insulin, or when the body cannot make good use of the insulin it produces". The body's failure to produce or properly use insulin results in heightened blood glucose levels ("high blood sugar"). This can in turn cause serious damage to the body and failure of various organs. Patients with diabetes have a higher risk of developing several serious health problems and diseases, such as cardiovascular diseases, blindness, kidney failure and lower limb amputation.

Types of diabetes

Diabetes is usually separated into 2 main categories; type 1 and type 2. In diabetes type 1 the body does not produce any insulin because of an autoimmune condition, meaning that the body attacks and destroys the insulin producing cells in the pancreas. Diabetes type 2 is a condition where the pancreas either produces too little insulin or where the

body is unable to properly utilize the insulin for intake of glucose to the cells (the latter is sometimes referred to as insulin resistance). The causes of diabetes type 2 are also somewhat unclear, but it is usually associated with obesity, dietary habits and genetics. [3]

Traditional treatment methods

Current medicine is unable to completely cure any type of diabetes. However, type 2 of the disease can often be managed with lifestyle changes such as increased physical activity and a healthy diet. Patients with diabetes type 1 are on the other hand always dependent on insulin treatment to avoid serious complications. The most common forms of diabetes treatment are insulin injections through syringes, insulin pens or insulin pumps. Several types of insulin are available for such treatments, usually categorized after how fast and for how long they act on regulating the blood glucose level. The four most common groups are rapid-acting, short-acting (”regular”), intermediate-acting and long-acting insulin.[4]

Insulin shots through syringe or pens need to be injected several times a day, and the patient must closely watch their blood glucose level, activity level, eating habits and injection timing in order to dose the right type and amount of insulin.

Insulin pumps are wearable devices which insert insulin to the body through a catheter, which is placed under the skin and attached to the pump by a tube. See figure 1.1.

Figure 1.1: Example of an attached insulin pump. [5]

These pumps deliver rapid- or short-acting insulin to the subcutaneous fat through the catheter. The pumps are preprogrammed to deliver a continuous stream of insulin at a so-called basal rate 24 hours a day. In addition to this, the user can manually add extra doses of insulin to manage rises in blood glucose levels for example when eating. These additional doses are called bolus doses. The insulin pumps do allow some level of automation, since the pump can be preprogrammed to deliver different basal rates at different times of the day (for example lower rates during the night). Still, traditional insulin pumps contain no feedback loops which regulate the insulin injection according to measurements of the actual glucose levels. Such pumps can therefore be characterized as pure open-loop sys-

tems, since the system input (amount of insulin injected) is completely predefined and not affected by measurements of the system response (blood glucose level). Figure 1.2 shows a graphic representation of an open-loop blood glucose control system. Because of this lack of automation, the user will have to manually keep a close watch over their blood glucose, meals, physical activity and make sure to add appropriate bolus doses. [4]

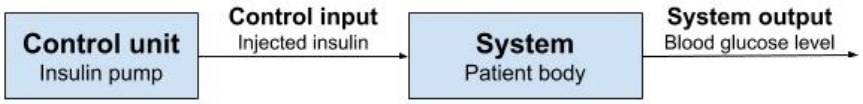

Figure 1.2: A traditional insulin pump works as a simple open-loop system.

Hypoglycemia and hyperglycemia

Hypoglycemia and hyperglycemia (popularly called low and high ”blood sugar”, respectively) occurs when a patients blood glucose levels diverge from normal. The conditions are characterized by a variety of symptoms, ranging from mild to severe. Severe hypoglycemia/hyperglycemia can lead to seizures, serious complications and even death. The most common cause for hypoglycemia is injection of insulin without a corresponding rise in blood glucose. [6] has the following advise for diabetes patients: ”If you take insulin or diabetes medicines that increase the amount of insulin your body makes – but dont match your medications with your food or physical activity – you could develop hypoglycemia.” Likewise, too low dosages of insulin relative to food intake could lead to hyperglycemia.

1.1.2 Artificial pancreas solutions

It is clear that even though diabetes treatment has come a long way, there is still a lot of inconvenience and a significant degree of risk present in the daily life of a diabetes patient today. Artificial pancreases are an evolution of the most commonly available insulin pumps on the market today. They are essentially a closed-loop version of the system displayed in figure 1.2, using real time measurements from the patient body to monitor and predict the blood glucose level. A diagram of an artificial pancreas system is shown in figure 1.3. Such systems will be able to adapt the insulin injections to the patient’s actual current needs, thus minimizing risk and removing the patients need to constantly monitor their own activities and glucose level.

Currently available artificial pancreases rely mostly on continuous glucose monitoring (CGM) as an input to the control unit. However, this is proving to be insufficient for accurate insulin regulation, since factors such as meals, stress, exercise and sleep patterns affect the glucose level in a rapid manner [7]. Insulin needs to be injected to the blood

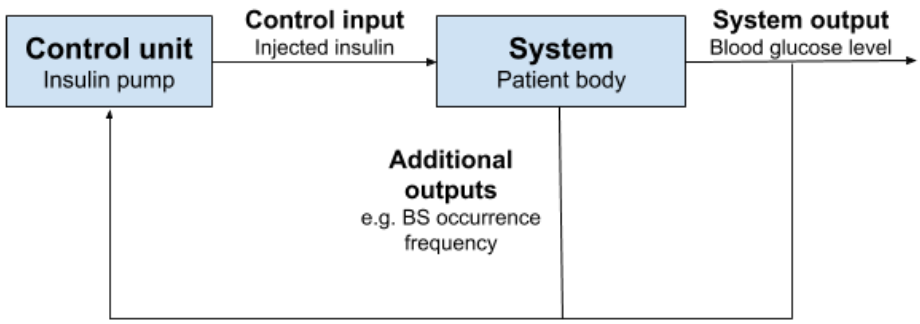

Figure 1.3: An artificial pancreas system closes the loop from fig 1.2 with continuous measurements of the patient's physical state.

stream some time before the blood glucose level starts to rise if it is to give any effect. Measurements of blood glucose levels alone are therefore not sufficient as an input to an automatic insulin regulation system, since the system would be too late with noticing the need for insulin injection. Thus, regulation of glucose levels in conjunction with meals is still a challenge in the field [8].

1.1.3 Artificial Pancreas Trondheim

Artificial Pancreas Trondheim is a research group which, as the name indicates, works towards the goal of developing an artificial pancreas system for diabetes patients. The goal is expressed at APT's website: "The long-term aim of APTs research is to develop a robust closed-loop glucose control system for patients with diabetes mellitus type 1 and 2 and for intensive-care patients, and to commercialize an artificial pancreas based on these results." [9]. The group was established in 2013 at NTNU in Trondheim, and consists of a multidisciplinary team of researchers.

1.1.4 Bowel sound detection in an artificial pancreas system

As mentioned in section 1.1.2, glucose monitoring alone is not sufficient in order to early enough detect a meal-related blood glucose increase. Other measurements will therefore be necessary, such that a meal can be detected right at it's onset, and the upcoming glucose increase can be predicted. One way of detecting a meal might be through monitoring bowel sounds. Section 2.1.3 refers to several sources which mention an increase in the number of bowel sounds after a meal, compared to when the patient is in a fasting state.

In order to continuously track the number of bowel sounds in a patient, it is first necessary to implement an automatic bowel sound detection system. Since the ultimate use of the detection algorithm merely will be counting bowel sound occurrences, a simple binary "bowel sound" vs "no bowel sound" detection scheme is sufficient. The algorithm will

thus not need to capture or distinguish between qualitative characteristics of the sounds. Development of such a detection system is the goal of this report.

1.2 Objective

1.2.1 Given problem

The delivered task from APT was stated as follows:

Norwegian version
Artificial Pancreas Trondheim (APT) jobber mot et langsiktig mål om å utvikle et robust automatisk glukosereguleringssystem for diabetikere. For å styre blodsukkeret hos diabetikere vil det være gunstig å detektere måltider så tidlig som mulig. Kanskje kan man gjøre det ved hjelp av lydmålinger fra magen, som vil si noe om hva som foregår i fordøyelsessystemet (på samme måte som "romling i magen", men kanskje utenfor det hørbare spekteret...). Slike lyder kan tas opp med et ganske enkelt stetoskop, liknende det man bruker for å lytte på lunger og hjerte, som er tilkoblet en PC for logging av data.

I denne oppgaven skal du:

1. Gi en beskrivelse av egenskapene til "bowel sounds" (tarmlyder), samt en fysiologisk beskrivelse av hva som forårsaker disse.
2. Gi en oversikt over relevante metoder innen signalbehandling av lyd.
3. Gjøre et begrunnet valg av hvilke metoder som best egner seg til å detektere diskrete hendelser i et datasett med lydmålinger, og implementere de valgte metodene p en relevant programvareplattform.
4. Teste og evaluere metodene valgt i pkt. 3 på et eksisterende datasett for å detektere tarmlyder.

English version
Artificial Pancreas Trondheim (APT) is working towards a long-term goal of developing a robust automatic glucose control system for diabetics. In order to control blood sugar levels in diabetics, it will be beneficial to detect meals as early as possible. Perhaps this could be done by means of sound measurements from the stomach, which can reveal something about what is happening in the digestive system. Such sounds can be recorded with a simple stethoscope, similar to what is used to listen to the lungs and heart, that is connected to a PC for logging of data.

In this task you should:

1. Describe the characteristics of bowel sounds, as well as a physiological description of what causes them.

2. Provide an overview of relevant methods of signal processing of audio.
3. Make a reasoned choice of which methods are best suited to detect discrete events in a data set of sound measurements, and implement the chosen methods on a relevant software platform.
4. Test and evaluate the methods selected in section 3 on an existing data set to detect bowel sounds.

1.2.2 Interpretation of the problem text

A problem text can always be interpreted in different ways depending on context and on who is reading it. This section will clarify the interpretation choices made when deciding how to approach the problem text in section 1.2.1.

The first point requires a physiological description of bowel sounds and what causes them. Since this is a technical report rather than medical, only a brief overview will be included here. The description will focus on the physiological aspects of bowel sounds that might be relevant for the further signal processing and classification problem. The report will also try to tie the physiological description together with observations made of the sound signal itself.

The second point states that an overview of relevant signal processing methods should be provided. The report will mention various signal processing methods found in literature related to detection and analysis of bowel sounds and other biomedical signals. The featured signal processing methods will be the ones considered relevant for the extraction of characteristic bowel sound traits.

Point number three calls for a selection of sound detection methods, and also an explanation of the choices made. A software implementation of the algorithm should also be made. After consultation with APT it was decided that an experimental ”proof of concept” implementation was sufficient. The implementation will therefore not consider any real time or efficiency aspects.

The final point requests a test and evaluation of the implemented detection algorithm on an actual data set of sound measurements. The data set(s) used for testing will be provided by APT, and this report will therefore not concentrate on the acquisition of sound measurements.

1.3 Approach

The first step of solving the given problem will be to examine the physiological properties of bowel sounds, as well as the properties of the sound signal itself. When the most prominent characteristics of bowel sounds have been determined, it is necessary to consider different signal processing methods which might be able to extract the different signal features. The extracted features will further be used as an input to a classifier, which will

use some pattern recognition scheme to differentiate between bowel sounds and surrounding noise or other sounds in the data set. Since there are several ways to build a classifier, a choice must be made on which pattern recognition method to use. Lastly, the complete algorithm containing feature extraction through signal processing and classification through pattern recognition will be implemented in code. Python was chosen as a suitable software platform, since it is open source and therefore easily accessible as well as rich in available libraries for signal processing and machine learning.

CHAPTER

2

BACKGROUND

2.1 Bowel sounds

2.1.1 Physiological description

Healthline.com defines bowel sounds as: ”noises made within the small and large intestines, typically during digestion.” [10] The sounds are usually caused by food, liquids and gas moving through the intestines by means of peristalsis, which are wavelike muscle contractions made by the intestines in order to transport substances through the digestive system. Most often bowel sounds arise due to regular digestion, and the sound activity normally increases in the period after the intake of a meal.

Bowel sounds can sometimes be heard with the naked ear (as ”stomach rumble”), but the majority of the sounds are inaudible unless you listen to the abdomen closely, for instance by using a stethoscope. The sounds vary greatly, but can generally be characterized as rumbling noises.

2.1.2 Acquisition of data

The provided data set from APT consisted of sound measurements from the stomach of one, self-declared healthy subject, who is a member of the APT research group. The subject ate a regular breakfast of müsli, fruits and milk about 5 hours before recording commenced. The subject could therefore be considered as being in a fasting state when measurements began. The equipment used for sound recording was a condenser microphone of type Sennheiser MKE2 P-C, which was inserted into the chest-piece of a classical

stethoscope, as shown in figure 2.1. The microphone was fixed in the upper right quadrant of the abdomen using medical tape. One-channel 24-bit audio signals with a sample frequency of 64 000 Hz were recorded. The signal was later downsampled to 4 000 Hz, and stored as a .wav-file.

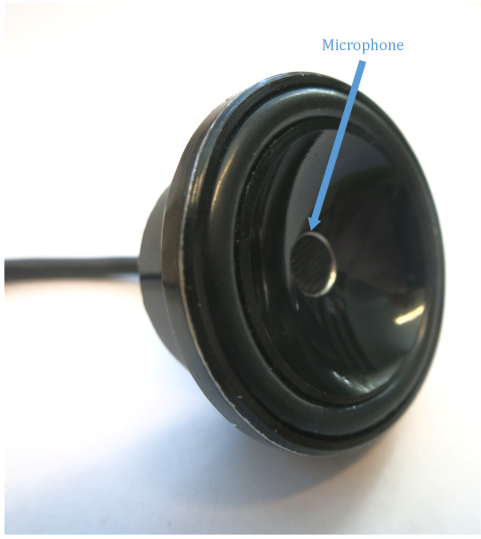

Figure 2.1: Chestpiece from a stethoscope with microphone in the center hole.

The recording protocol ensured that bowel sounds were recorded during fasting, eating and digestive states. As mentioned, recording started in fasting state. After 51 minutes, the subject started to eat a meal consisting of bread with cheese, an activity which lasted about 15 minutes. The meal was followed by a 45 minute-lasting digesting period. The recording ended after 90 minutes in total [11].

During the recording, the subject remained reclined and tried to move as little as possible. Despite the subject's best effort, uncontrollable events such as sneezing still occurred during recording. These data segments were still included in the data set used for detection, in an attempt to design the detection algorithm such that it can distinguish bowel sounds from other interfering sounds.

2.1.3 Bowel sound signals

Auscultation of bowel sounds has been widely used to observe the motility and general workings of our digestive system, ever since the pioneering works of Cannon [12]. However, the analysis of bowel sounds is still done manually and quite subjectively, due to the lack of consent on quantitative characteristics of bowel sounds [13].

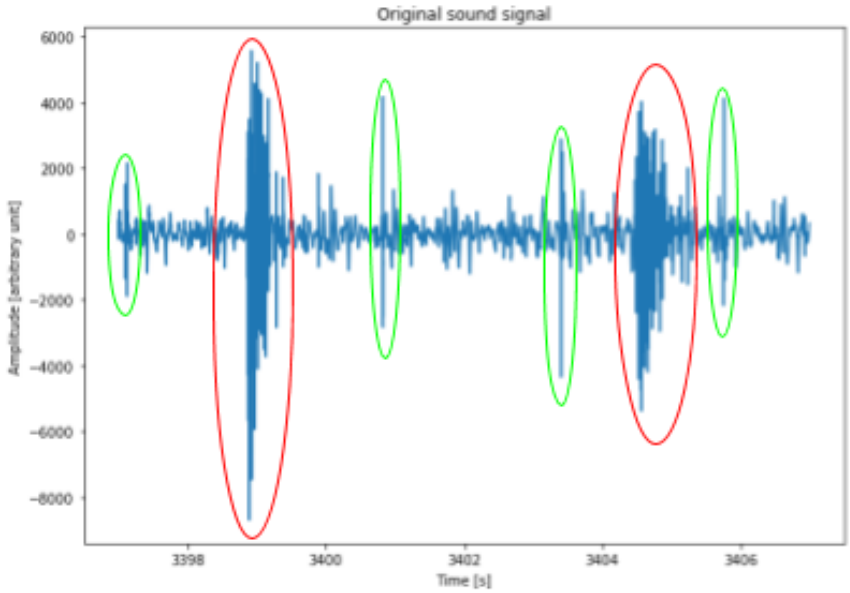

Figure 2.2: Example of an untreated bowel sound signal over the duration of 10 seconds. Clear growling or rumbling noises are marked in red, more faint clicks or taps marked in green.

A first look on the signal

A typical example of a bowel sound signal plot is shown in figure 2.2, which is a portion of the data set received from APT. As can clearly be seen, the signal is contaminated by a significant amount of noise. The bursts marked by red circles are clearly distinguishable bowel sounds similar to growling or rumbling. The spikes marked by green circles are heard as clicks or taps, but it is hard to tell if these are actual bowel sounds or artifacts stemming from for instance friction from clothes. However, concentrating on the more obvious bowel sounds (red circles), it is clear from visual inspection that the sounds are characterized by increased amplitude and frequency compared to the background noise. Apart from these very general trademarks, the appearance of a bowel sound signal varies greatly from event to event.

Quantitative description

These observations of heightened amplitude and frequency, but also of high irregularity and significant noise contamination, agree well with the descriptions of bowel sound signals found in literature. Table 2.1 contains an overview of bowel sound characteristics found in different studies. Sound signals are in general described by indexes such as frequency spectrum, duration and amplitude. Still, as described earlier, there is no general consensus on the characteristics of bowel sounds, and quantitative descriptions vary somewhat from study to study.

Source	Dominant frequency range of BS (Hz)	BS duration (ms)
[14]	100 - 1000	5 - 200
[13]	225 - 530	200 - 1500
[15]	100 - 500	20 - 5000
[16]	240 - 300	5 - 10
[17]	150 - 450	10 - 100

Table 2.1: Bowel sound (BS) characteristics found in literature. Approximations have been made to some of the data, since only rough range estimates are necessary for this report.

When looking at the studies [14; 13; 15; 18; 16; 17] collectively, it seems fair to assume frequency ranges between 100 Hz and 500 Hz for the signal portions containing bowel sounds. The same sources also all agree that a portion of the superimposed noise stems from cardiovascular and respiratory activity, and thus has a frequency range much lower than those of bowel sounds, typically below 80 Hz. Nonetheless, there is a significant amount of noise whose frequency spectrum overlaps with the spectrum of the bowel sounds themselves.

Reports on sound durations are less consistent, and sounds should be expected to last as short as a few milliseconds or possibly as long as several seconds. This large divergence in duration ranges could come from differences in how each study defines a single bowel sound. For instance, [17] defines ”clusters”, which are ”two sounds less than 15 msec apart or sounds that are so close that they are merged and counted as a sound with a duration of greater than 90 msec”. An example of such a phenomenon can be found in figure 2.3. If counted as one single sound, the duration of this sound event would be around 1 second. However, if viewed as many consecutive sound events, each sound only has a duration of a few milliseconds.

Quantitative values of amplitude are even harder to find a common consesus for. The absolute value of the sound amplitude varies greatly according to choice of instrumentation, placement of microphone/stethoscope with respect to the sound’s place of origin, amount of fat/muscle beneath the skin etc. It is therefore irrelevant to mention any specific amplitude values, but all studies agreed that the amplitude of a bowel sound is high relative to the surrounding background signal.

Bowel sound signals in relation to food intake

Several studies have examined how bowel sounds behave and possibly change throughout the process of digesting a meal. [15] concludes that both sound intensity (amplitude), duration and frequency can be used as indexes to examine evolution of a digestion cycle, but the most significant index is the number of sounds detected per minute. [17] found that sound duration and intensity remained quite similar in both fasting and fed state, while the sound occurrence frequency increased drastically. [19] measured an increase of 1.5

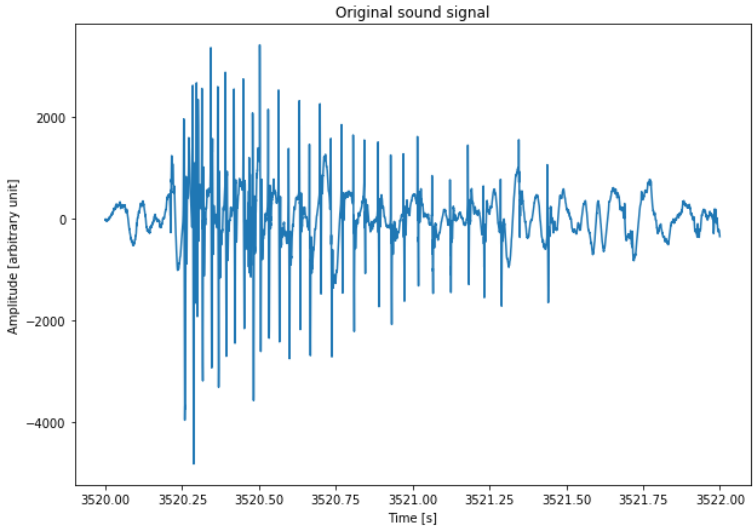

Figure 2.3: Example of a cluster of short bowel sounds, which can also be viewed as a single long sound.

times as many occurrences of bowel sounds directly after eating, compared to when the test subject was in a fasting state.

These findings are all in compliance with the physiological descriptions of the origins of bowel sounds from section 2.1.1. Eating will both lead to increased peristaltic activity and to increased amounts of contents in the intestines. This will again lead to an increased occurrence frequency of bowel sounds.

2.2 Signal processing methods

2.2.1 Frequency analysis

Fourier transform

The idea that periodic functions can be decomposed into sines and cosines was first systematically employed by Jean-Baptiste Joseph Fourier in his work with heat conduction in 1822 [20]. Periodic signals can be reconstructed completely by taking a linear combination of sines and cosines with different frequencies and amplitudes and combining these to an infinite series called a *Fourier series*. The generalization of Fourier series is obtained when we let the signal period approach infinity (in practice this corresponds to a non-periodic function), such that Fourier analysis can be applied also to non-periodic functions. This analysis method is called the *Fourier transform* of a function, and yields the frequency spectrum of a signal. The frequency spectrum is a representation of the signal which displays signal amplitude with respect to frequency, and thus belongs to the

frequency domain rather than the time domain. Simply explained, one could say that the frequency spectrum shows "how much" content a signal has of each frequency. This representation can be useful in several engineering applications, for instance in noise filtering and frequency analysis.

The equation for the Continuous Fourier Transform (CFT) is given in equation 2.1 [21, p. 242], where $X(F)$ is the transformed version of the input function $x(t)$ and F is the frequency variable.

$$X(F) = \int_{-\infty}^{\infty} x(t)e^{-i2\pi Ft}dt \tag{2.1}$$

Discrete Fourier Transform and the Fast Fourier Transform

The original version of the Fourier transform is valid for continuous aperiodic signals. However, since most computations today are done by computers, which can only handle discrete signals and values, it is necessary to use a discrete version of the transform. In the Discrete Fourier Transform (DFT), both the time and frequency domain are discretized into N equally sized intervals (in the time domain) or bins (in the frequency domain), such that both input and output can be handled digitally.

A continuous real-valued signal will contain both positive and negative frequencies, and it's spectrum is mirrored around 0 Hz. If a real signal is sampled at frequency f_s, it's spectrum will appear periodic with period f_s, and the symmetry around 0 Hz will reappear at every frequency $n\frac{f_s}{2}$ (n is any integer) [21, p. 271]. This is provided that the sampling frequency is chosen according to the Sampling Theorem (Nyquist's rule) [21, p. 30], such that no aliasing occurs. Figure 2.4 illustrates the spectra of a real, continuous signal and the same signal correctly sampled with sampling frequency $f_s \geq 2B$. From the figure it becomes clear that we can remove everything from the bottom spectrum except from the range $f = [0, f_s/2]$ without losing any frequency information. This is usually done when plotting the spectrum of a real, digital signal, and such a plot is called the *one-sided frequency spectrum*.

Equation 2.2 [21, p. 256] describes the Discrete Fourier Transform, where $X(\omega)$ is the transformed version of the input function $x[n]$, n is the discrete time variable, ω is the discrete frequency variable and N is the total number of samples.

$$X(\omega) = \sum_{n=0}^{N-1} x[n]e^{-2\pi i\omega n/N} \tag{2.2}$$

The Fast Fourier Transform (FFT) is simply an algorithm for efficiently calculating the DFT of a signal. This is the algorithm most commonly used when digitally computing a frequency spectrum.

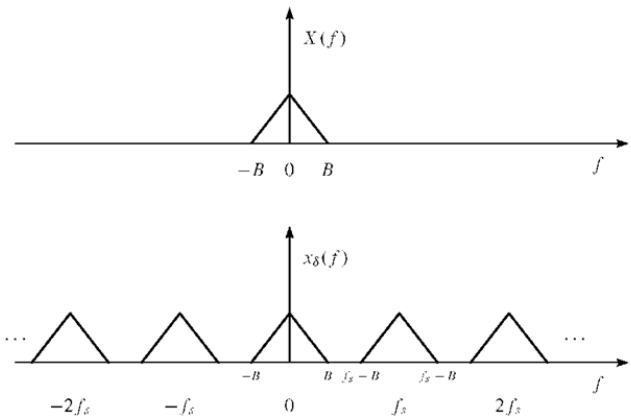

Figure 2.4: Illustration of the frequency spectra of a continuous signal with bandwith B (top) and the signal sampled with sampling frequency f_s (bottom). The one-sided spectrum for this signal would be the portion between $f = 0$ and $f = B$. [22]

Power Spectral Density

A power spectral density (PSD) plot displays how the energy of a signal is distributed among the frequency components found in the signal. A function describing signal power per unit frequency is found in equation 2.3 [21, p. 903], where S_{xx} is the power spectral density function $X(F)$ is the Fourier transform of the signal $x[n]$ and N is the total number of samples of x.

$$S_{xx}(\omega) = \frac{1}{N}|X(\omega)|^2 \tag{2.3}$$

Since most real-life signals are stochastic processes which goes on for infinity, they do not have a finite total power. Equation 2.3 is therefore only an estimate for the true spectral power of the signal, estimated from a collection of samples. This specific estimate is named a *periodogram* [21, p. 903]. In order to better account for noise and sharp fluctuations in the data, a windowing estimation method might be used to better approximate a signal's actual PSD. Welch's method splits the data into overlapping segments, calculates a periodogram for each segment using DFT, and then averages over all periodograms in order to obtain a smoother estimate of the PSD.

Wavelet Transform

The Uncertainty Principle

The most common forms of frequency analysis are based on different variations and adaptions of the classic Fourier transform (see section 2.2.1), which is an integral transform. Although efficient for extraction of frequency content, these methods all have restrictions when it comes to localizing frequency information exactly in time. This phenomenon

stems from the so-called *Uncertainty Principle* (also called the Heisenberg-Gabor inequality) [23, p. 56]. Basically, this principle shows that it is impossible to determine a non-stationary signal's exact frequency at an exact point time. The extreme version of this is classical Fourier transformation, which outputs a frequency-amplitude representation with no time resolution. This means that the transform determines which frequencies exist in a given signal, but has no information about where these frequencies occur in time. Other approaches to frequency analysis that are not based on integral transforms do exist, and these will not suffer from the limitations of the Uncertainty Principle [24]. However, such methods will not be discussed in this report.

The Wavelet Approach

The wavelet transform is a sort of work-around for this time-frequency resolution problem. The transform uses a basis function, a *wavelet*, which holds certain waveform characteristics, but is limited in time. A wide selection of wavelet forms exist, and a suitable wavelet has to be picked for each application. The wavelet can be scaled so that the width of the wavelet widens or narrows. The wavelet will be translated along the signal to be analyzed several times, each time with a different scale. For each translation the correlation between the wavelet and the underlying signal is calculated, thus extracting the frequency components. When a small-scale wavelet is used (narrow width), the high frequency components will be obtained, while the larger scales (large width) will extract lower frequencies. The scale and translation pairs will correspond to time and frequency, thus obtaining a sort of time-frequency representation of the original signal. The method is similar to the *Short-time Fourier transform* (STFT), but has certain advantages. STFT, a windowed version of Fourier, will use a fixed window size for the frequency analysis, thus yielding a static resolution in time-frequency space. The wavelet transform adapts it's window according to the scale, and thus yields higher time resolution for high frequency signals.

2.2.2 Signal filtering

A very general description of a filter is a device that receives some sort of input and only allows a specific portion of this input to pass through to the output. In the case of signal processing the input is a signal, and the filter is most often used to suppress or block out certain frequency components. The filter can be realized physically, for instance with components in an electric circuit, or it can be created digitally by implementing the filter's associated mathematical functions or algorithms in code. A filter's associated functions are a model of the filter's behaviour, and thus decides the relationship between the input and output signal. The filters used in this assignment can all be modeled as linear time-invariant systems (LTI systems). This means that the functions which describe the filter's input-output relationship are all both linear and time-invariant. The filter's modification of the input signal can be expressed in terms of the system transfer functions in the following way: "In general, a linear time-invariant system modifies the input signal spectrum $X(\omega)$ according to its frequency response $H(\omega)$ to yield an output signal with spectrum $Y(\omega) = H(\omega)X(\omega)$." [21, p. 330]

Highpass and lowpass filters

Filters that allow only the high- or low-frequency components of a signal to pass through to the output are called highpass filters and lowpass filters, respectively. The threshold distinguishing between which frequencies should pass and which should be suppressed is called the filter's *cutoff frequency*. Ideal versions of such filters will output a non-distorted version of the input signal, with unity gain within the passband, and zero gain within the stopband. A plot of the amplitude (gain) response of an ideal filter is found in figure 2.5. From the figure we see that the output amplitude is multiplied by 1 for all frequencies within the passband, and by 0 for all frequencies belonging to the stopband portion of the frequency spectrum.

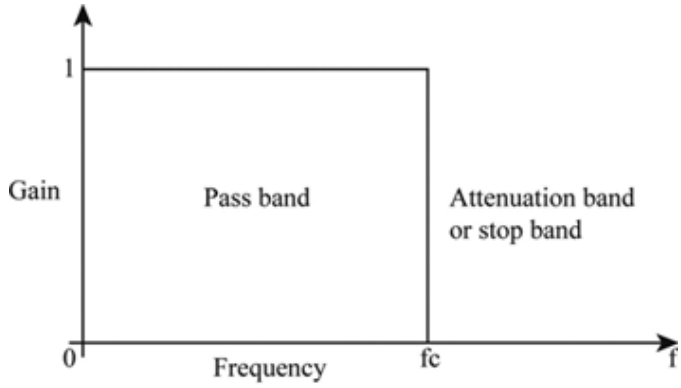

Figure 2.5: Amplitude response of an ideal lowpass filter with cutoff frequency f_c [25]

Butterworth filter

Ideal filters are unfortunately not possible to implement in real life. However, certain filter types can come very close to some of the ideal filter characteristics. A Butterworth-filter is designed from the principle that "an ideal electrical filter should not only completely reject the unwanted frequencies but should also have uniform sensitivity for the wanted frequencies" [26]. A Butterworth-filter will therefore yield a unity gain in almost the entire passband of the filter, and then a gain decreasing towards zero as we move into the stopband. Compared to other filter types, the desirable flat frequency response in the passband comes at the expense of the steepness of the cutoff curve. The gain response of a lowpass Butterworth filter is shown in figure 2.6. The cutoff is not as sharp as for the ideal filter in figure 2.5, but gets sharper as we increase the order of the filter. A higher order filter gives a better amplitude preservation in the passband, and a sharper cutoff curve to the stopband. However, very high filter orders make the total system more complex, which for a digital implementation can result in longer computation times. It is also worth noting that the Butterworth filter has a phase response which will shift and slightly distort the phase of the output signal in comparison to the input signal's phase.

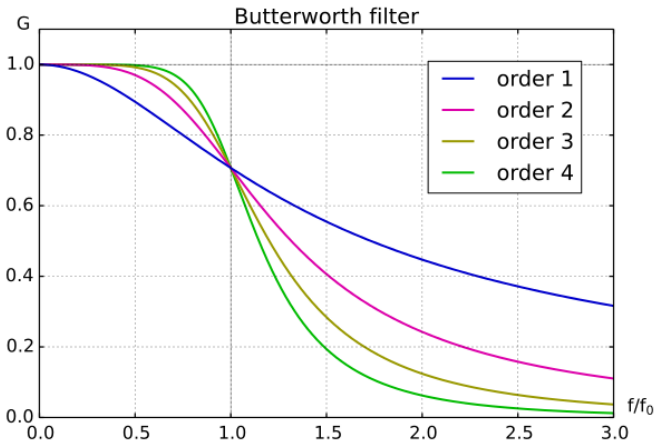

Figure 2.6: Amplitude response of Butterworth lowpass filters of different orders [27]

2.2.3 Hilbert transform

A Hilbert transform is used to acquire the analytic representation of a signal, such as in equation 2.4, where $A(t)$ is the instantaneous amplitude and $w(t)$ is the instantaneous frequency. It is worth noting that this calculation of instantaneous frequency only is correct for a limited class of signals, and would not yield meaningful results for a multicomponent, non-stationary signal such as the signal to be analyzed in this report [28]. The variable $w(t)$ will therefore not be used further.

$$z(t) = A(t)e^{jw(t)} \tag{2.4}$$

The complex-valued analytic function $z(t)$ is created from a real-valued function $x(t)$ by adding the Hilbert transformed signal as the imaginary part, as shown in equation 2.5. $HT(\bullet)$ here denotes the Hilbert transform.

$$z(t) = z_r(t) + jz_i(t) = x(t) + jHT(x(t)) \tag{2.5}$$

This representation of a signal is often used to obtain the signal envelope, such as depicted in figure 2.7. The envelope curve is found by plotting the magnitude of the analytic signal $z(t)$ for each time instant, or in other words, plot the curve $A(t)$.

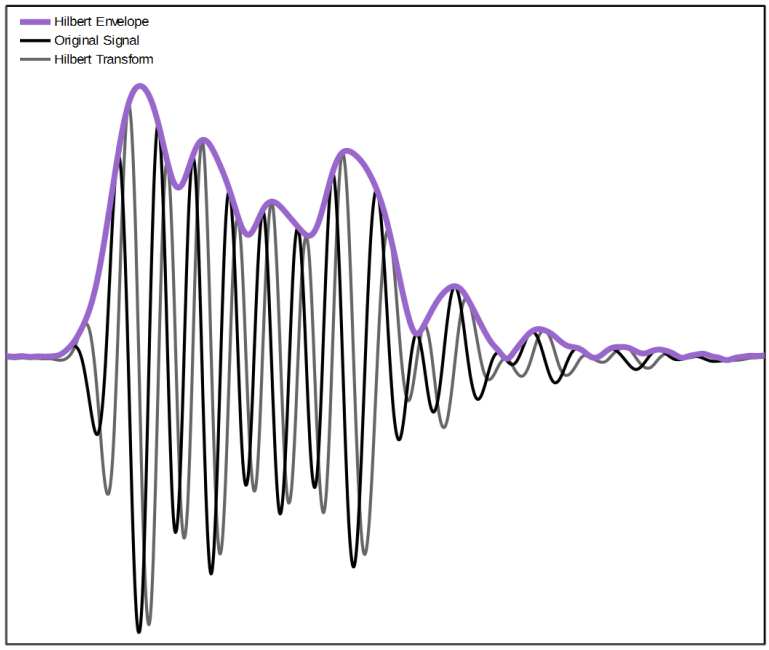

Figure 2.7: Signal envelope obtained from a Hilbert transform. [29]

2.3 Pattern recognition

Pattern recognition is the act of recognizing a certain pattern or structure in a set of raw input data, and is becoming an increasingly important part of modern automated systems. When implementing pattern recognition algorithms in digital systems, the goal is very often that of *classification*, namely to sort the raw data into a certain number of discrete classes or categories. Throughout section 2.3 the material will be based on [30] unless otherwise stated.

2.3.1 A generic classification system

A diagram explaining a generic classification system is depicted in figure 2.8. Appropriate sensors will measure relevant information about the physical phenomenon that we wish to classify and represent this as some form of raw data. The system will then extract predefined features from the raw data, which better represent the differences between the classes. The set of feature values is then passed on to the classifier, which contains some decision rule which takes a vector of feature values as an input. The output from the decision rule is which class the phenomenon in question belongs to (according to our classification system).

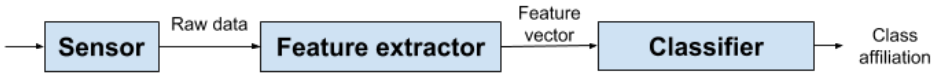

Figure 2.8: An overview of a generic classification system.

Discriminant functions

Decision rules can be simplified and generalized if they are formulated with *discriminant functions*. These are a set of functions $g_i(\mathbf{x}), i = 1, ..., c$, one for each class i in the classification problem, which take a feature vector $\mathbf{x}$ as input. When correctly designing such discriminant functions, the decision rule for a classification problem can be expressed as equation 2.6.

$$\text{Choose } \omega_i \text{ if } g_i(\mathbf{x}) = \max_j \{g_j(\mathbf{x})\}. \tag{2.6}$$

For a two class problem, this equation can be simplified futher by combining the two discriminant functions such that $g(\mathbf{x}) = g_1(\mathbf{x}) - g_2(\mathbf{x})$. This yields the decision rule in equation 2.7.

$$\text{Choose } \omega_1 \text{ if } g(\mathbf{x}) > 0, \ \omega_2 \text{ otherwise.} \tag{2.7}$$

Training a classifier

The method behind constructing an effective decision rule, through a set of discriminant functions, is the main topic of the field pattern recognition. The functions will be determined based on some data set with which the classifier is *trained*. During training, the statistical properties of the data set is examined, and structures or patterns are sought. The result is an unambiguous discriminant function which will classify new, unseen data samples. An illustration of the two phases, training and usage, can be found in figure 2.9. The training or fitting stage will usually consist of an iterative process of training, testing and tuning of the classifier before a final decision rule is obtained.

Supervised and unsupervised learning

Methods for training a classifier can be divided into two categories, *supervised* and *unsupervised* methods. In supervised methods, the data set given to the classifier for training also contains a corresponding set of reference values with the correct class affiliation for each feature vector. Providing these reference values, or *marking* the data set can sometimes be too time consuming or simply impossible. In such cases, unsupervised methods might be used. Here, the classifier attempts to arrange the data set into groups with similar feature vectors. How to define the vectors as ”similar” varies from method to method, where different approaches are suitable for different applications.

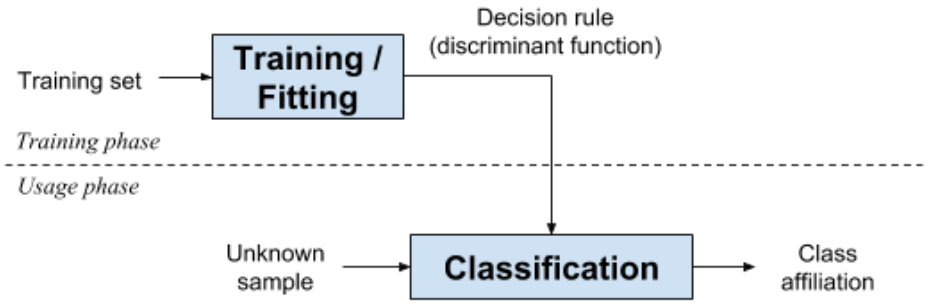

Figure 2.9: Training and usage of a classification system. Figure inspired by [31, p. 9]

Parametric and nonparametric methods

Some training methods make certain assumptions about the statistical properties of the data when constructing discriminant functions for a classifier. Typically, the probability density distribution of the features within each class will be assumed to follow some known distribution function. The task to be solved during training is then to find the correct parameters that fit the chosen distribution function to the given training data. Such approaches are called *parametric methods*. In contrast, methods who make no assumptions about the statistical properties of the data are called *nonparametric methods*.

2.3.2 Unsupervised learning methods

Clustering methods

Clustering is a collection of unsupervised machine learning techniques which attempt to gather the samples of a data set into subgroups of similar samples. Similarity between samples is very often measured by the distance between samples when they are plotted in feature space. Different clustering algorithms use different distance measurements.

K-means

One of the most common clustering methods is the *K-means* method. This procedure is efficient also for large data sets. The k-means algorithm is described by algorithm 1. The algorithm takes the number of clusters k as an input parameter, together with the samples $\mathbf{x}_i,\ i = 1, ..., n$. The algorithm initializes points in feature space $\boldsymbol{\mu}_j,\ j = 1, ..., k$ (not from the data set, even though $\boldsymbol{\mu}_j$ and $\mathbf{x}_i$ might coincide), which are called the *centroids* of the clusters. These centroids are the mean value of all the samples within each cluster, and thus represent the ”heart” of each category. Throughout the execution of the algorithm, the centroids will be moved around with the goal of minimizing the distance from each sample $\mathbf{x}_i$ to it's respective cluster centroid $\boldsymbol{\mu}_j$. When no improvement of the clustering can be done, or when a maximum number of iterations are met, the algorithm

terminates. The results are the k centroids $\boldsymbol{\mu}_j$, which hold the mean values of each cluster. Each sample $\mathbf{x}_i$ will be labeled with the class label of the nearest centroid.

Algorithm 1: The k-means algorithm.

Input: $k, \mathbf{x}_1, ..., \mathbf{x}_n$
Output: $\boldsymbol{\mu}_1, ...\boldsymbol{\mu}_k$

```
begin Initialize μ_1, ...μ_k
    repeat
        classify n samples to nearest μ_j
        recompute μ_j
    until no change in μ_j;
end
```

2.3.3 Supervised learning methods

Linear discriminant analysis

Linear discriminant functions are easy to compute, and are therefore suitable as an initial classifier candidate. While some classification methods make assumptions about the features' probability distribution within each class, discriminant analysis rather assumes a known form of the discriminant function. In this sense, the procedures can be said to be nonparametric.

Machine learning procedures based on linear discriminants assume that the classification problem can be solved with discriminant functions of the form $g_i(\mathbf{x}) = \mathbf{w_i^T x} + w_{0i},\ i = 1, ..., c$. For a two-class case, the discriminant functions can again be combined into one single function, as seen in equation 2.8. Here, $\mathbf{w} = \mathbf{w}_1 - \mathbf{w}_2$ and $w_0 = w_{01} - w_{02}$. This allows us to use the decision rule from 2.7.

$$g(\mathbf{x}) = \mathbf{w^T x} + w_0 \tag{2.8}$$

Since the sign of $g(\mathbf{x})$ decides the class membership of a sample, $g(\mathbf{x}) = 0$ is the boundary which separates the two regions in feature space containing samples from ω_1 and ω_2, respectively. Geometrically, $g(\mathbf{x}) = 0$ describes a hyperplane, where $\mathbf{w}$ is the normal vector to the plane and thus determines it's orientation. The value of $g(\mathbf{x})$ is then the signed distance from a sample $\mathbf{x}$ to the hyperplane. Classifier training consists of locating the hyperplane such that it correctly separates the two classes, based on the information given from the training samples.

To simplify the problem of finding a correct hyperplane equation, $g(\mathbf{x})$ is mapped from the original feature space ($\mathbf{x}$-space) onto a higher dimensional $\mathbf{y}$-space by augmenting the vectors $\mathbf{x}$ and $\mathbf{w}$ as shown in equation 2.9.

$$g(\mathbf{x}) = \mathbf{w^T x} + w_0 = w_0 + \sum_{i=1}^{d} w_i x_i = [w_0, w_1, ..., w_d] \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix} = \mathbf{a^T y} \tag{2.9}$$

In $\mathbf{y}$-space, the hyperplane has no bias w_0, and therefore runs through origo. The optimization problem now consists of tuning the weight coefficients of $\mathbf{w}$ in order to orientate the hyperplane so that all training samples are situated on the correct side of the plane (positive side for ω_1 and negative side for ω_2). This problem can be solved by minimizing an appropriate loss function by using a gradient descent algorithm.

Linear separability

In order for a linear discriminant classifier to work perfectly, the data set to be classified need to be *linearly separable*. A linearly separable set is shown in figure 2.10. It is clear that in this case the data can easily be separated by a linear hyperplane (in this case, a one dimensional line).

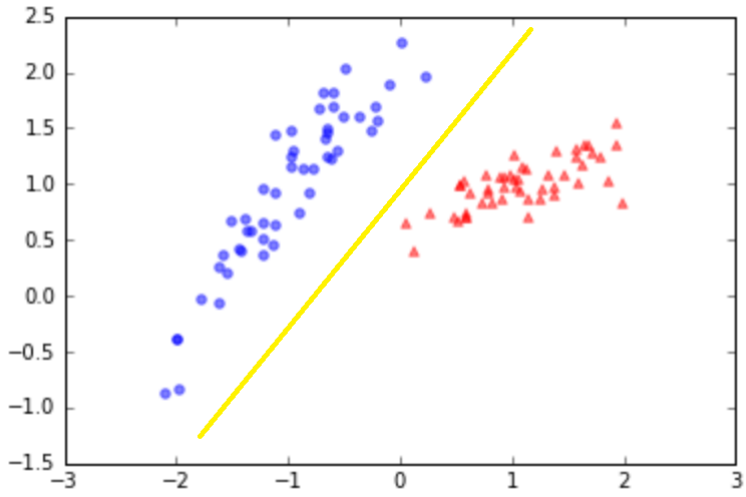

Figure 2.10: A linearly separable data set.

It is not always as easy to visually determine linear separability as for the case in figure 2.10. Generally, "with an appropriate nonlinear mapping to a sufficiently high dimension, data from two categories can always be separated by a hyperplane." [30, p. 259]. This property is exploited by kernel support vector machine algorithms, but this is outside the scope of this report. The important point to notice from this fact is that a data set might be

separable in a dimension d, but might lose this property when projected onto a space with dimension $\hat{d}$, $\hat{d} < d$. An example of this situation is illustrated in figure 2.11. Since data visualization in dimensions higher than 3 often is difficult, linear separability might need to be determined in other ways.

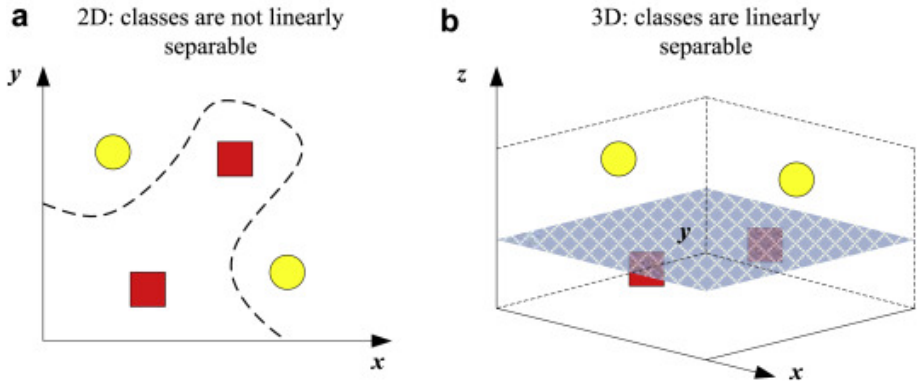

Figure 2.11: A data set which is linearly separable in 3 dimensions, but not 2.

Linear Support Vector Machines

A specific type of linear discriminant classification is a linear *support vector machine* (SVM). This algorithm seeks to find a class-separating hyperplane such that the distance from the hyperplane to the nearest samples from each class (called the *support vectors*) is as large as possible. Figure 2.12 illustrates an optimal hyperplane resulting from training an SVM. The distance from the hyperplane to the support vectors is called the *margin*.

The original version of an SVM (called *hard margin* SVM) requires all training samples to be on the right side of the margin. However, outliers or noise in the data set will then greatly influence the location of the hyperplane as well as the size of the margin. This problem, called *overfitting*, will lead to a classifier which is very good at classifying the samples of the training set, but not so good at classifying new and unseen samples. Overfitting will be further discussed in section 2.3.4. A remedy for this issue is to use the *soft margin* version of SVM. This algorithm use a loss function called *hinge loss*, which is described by equation 2.10, to search for an optimal hyperplane.

$$L(\mathbf{y}, z) = \max\{0, 1 - zg(\mathbf{y})\} \tag{2.10}$$

Here, $\mathbf{y}$ is the augmented feature vector, z is the target value of vector $\mathbf{y}$ mapped to $\mathbf{y}$-space and $g(\mathbf{y})$ is described by equation 2.9. The mapped target value z will take on the value $z_k = \pm 1$, depending on whether sample k is in class ω_1 or ω_2. Since $g(\mathbf{y}_k)$ gives the signed distance from a sample $\mathbf{y}_k$ to the hyperplane, the term $1 - zg(\mathbf{y}_k) < 0$ when g predicts the correct class for $\mathbf{y}_k$, and the distance from $\mathbf{y}_k$ to the hyperplane is at least 1. This situation will result in 0 loss. Otherwise, L will output the distance from $\mathbf{y}_k$

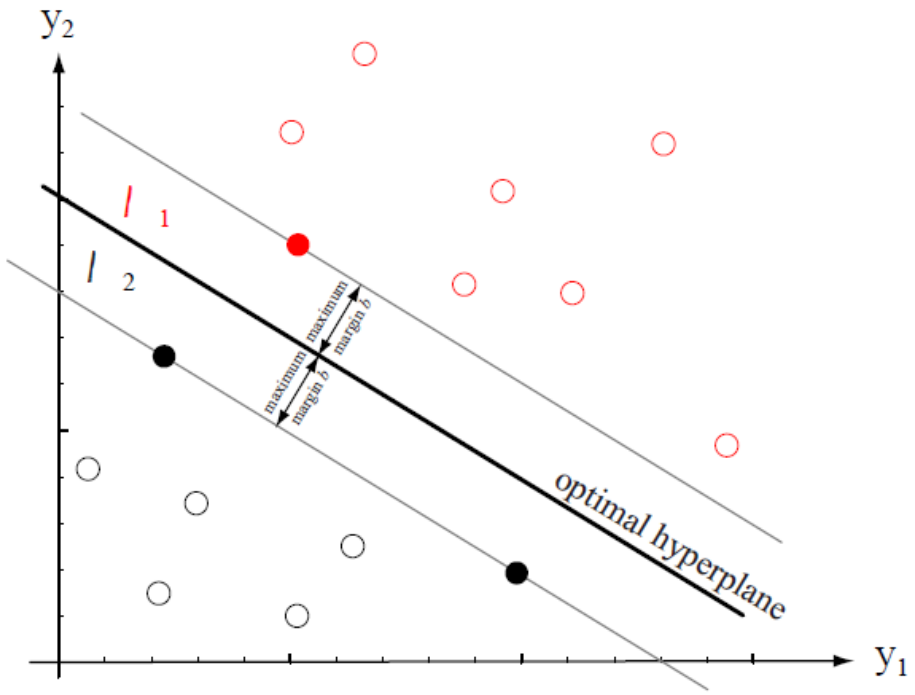

Figure 2.12: The goal of a support vector machine is to find a separating hyperplane that maximizes the margin. Support vectors are shown as solid dots. [30, p. 263]

to the right side of the margin. The minimal solution of equation 2.11 [32] thus locates the hyperplane such that each training sample is sufficiently close to the right side of the margin. The term $\lambda\|\mathbf{a}\|^2$ is a regularization term which allows the user to adjust the trade-off between low classification error in training samples and generality of the classifier for new samples by adjusting the factor λ.

$$\min_{\mathbf{a}} \left[\frac{1}{n} \sum_{k=1}^{n} \max\left(0, 1 - z_k(\mathbf{a}^{\mathbf{T}}\mathbf{y})\right) + \lambda\|\mathbf{a}\|^2 \right] \tag{2.11}$$

2.3.4 Evaluation of classifiers

Cross-validation

A common mistake when training a classifier is to test the performance of the classifier on the same data set that was used for the training. This will in most cases lead to a classifier that perfectly classifies the training samples, but performs poorly on new samples. This problem is referred to as *overfitting* of the model.

To avoid overfitting, it is important to save a portion of the data set to test the classifiers performance, and not use this test set in the training procedure. To further make sure that the classifier is not just tweaked to perform well on the test set, a third validation set might also be set aside for final evaluation after the final classifier parameters are obtained. The process of setting aside a subset of the data for testing and validation is referred to as *cross-validation*.

Classifier evaluation metrics

A *confusion matrix* is often computed during the evaluation process of a classifier. This matrix summarizes how ”confused” your classifier is while making predictions. For a binary classification problem, the matrix will look like the one presented in table 2.2 [33].

		Predicted class	
		0	**1**
True class	**0**	Number of true negatives	Number of false positives
	1	Number of false negatives	Number of true positives

Table 2.2: Confusion matrix for two-class classification.

From the confusion matrix, a number of metrics can be computed. The *sensitivity* is the true positive rate, calculated by equation 2.12. In the bowel sound context, this measures a classifiers ability to correctly predict the class ”bowel sound” for samples which are actually bowel sounds.

$$\text{sensitivity} = \frac{\text{true positives}}{\text{true positives + false negatives}} \tag{2.12}$$

The *specificity* is the true negative rate, expressed in equation 2.13. This metric measures the classifiers ability to correctly predict the class ”no bowel sound” for the samples which are in fact background signal.

$$\text{specificity} = \frac{\text{true negatives}}{\text{true negatives + false positives}} \tag{2.13}$$

A common estimate for classifier performance is the *error rate*, given in equation 2.14. This is the rate of wrongly classified samples, regardless of true class label.

$$\text{error rate} = \frac{\text{false positives + false negatives}}{\text{all samples}} \tag{2.14}$$

The *precision*, also called *positive predictive value* is found in equation 2.15. This equation calculates the ability of the classifier not to label as positive a sample that is negative.

$$\text{precision} = \frac{\text{true positives}}{\text{true positive + false positive}} \tag{2.15}$$

Lastly, the rate of total *accuracy* is given by equation 2.16. This is the rate of correctly classified samples, regardless of true class label.

$$\text{accuracy} = \frac{\text{true positives + true negatives}}{\text{all samples}} \tag{2.16}$$

2.4 Feature selection

When building a pattern recognition application, the features chosen as inputs to the classifier algorithm are often chosen because they are believed to represent a clear distinction between the different classes of the problem. Nonetheless, the initial set of features might turn out not to be the optimal set. In some cases a subset of the initial features might yield a better performance from the classifier than the total set, as a too large set of features might lead to *overfitting* (see section 2.3.4) [34]. In addition, it is clearly more computationally efficient to work with a smaller set of features. It is therefore recommended to make a quantitative evaluation of the original feature set such that a more optimal set of features might be extracted.

2.4.1 Recursive feature elimination

Recursive feature elimination (RFE) is an algorithm for selection of the optimal feature subset from a superset of original features. The method is presented in algorithm 2 [34]. Here, n is the number of samples in the training set, F_0 is the initial number of features and F is the desired number of features in the final subset.

Algorithm 2: Recursive feature elimination (RFE) algorithm for a generic classifier.

Input: Training feature vectors: $\mathbf{X}_0 = [\mathbf{x}_1, ..., \mathbf{x}_n]$, Sample labels: $\mathbf{y} = y_1, ..., y_n$
Output: Subset of surviving features $\mathbf{s} = [1, ..., F]$

begin Initialize: subset of surviving features $\mathbf{s} = [1, ..., F_0]$
  repeat
  Restrict training vectors to surviving features: $\mathbf{X} = \mathbf{X}_0[:, \mathbf{s}]$
  Train the classifier
  Compute the feature scores
  Remove feature(s) with lowest score(s) from $\mathbf{s}$
  until $\mathbf{s} = [\,]$
end

Given a training set and a classifier, the algorithm recursively eliminates badly performing features from the surviving subset until the desired number of features are left. The

performance of the features are measured by how important they are for the classifier to correctly classify the training samples. This measure of importance will vary from classifier to classifier.

Recursive feature elimination with cross validation

The RFE algorithm can be further extended to also contain *cross validation* of how the surviving feature subsets perform in the given classifier. The cross-validation technique is further explained in 2.3.4. In this modified version, the algorithm tests the performance of the classifier for each new feature subset by means of cross-validation. This process of feature elimination and evaluation is repeated until all features have been removed from the surviving set. Then, the subset with best performance is selected. In this way, the algorithm itself will calculate the optimal number of features to retain in the final feature set [35].

The performance measurement for the cross-validation scheme will vary with the application needs. For example the classifier accuracy can be used, which is described by equation 2.16.

2.5 Python

Python is a high-level, open source programming language with great prevalence in the global informatics and engineering community today [36]. The language emphasizes code readability, and is therefore an ideal language for beginners. It's open-source nature encourages development of libraries for a wide range of use-cases, and facilitates cooperation and sharing of knowledge on forums across the internet. The broad selection of libraries makes it easy to realize originally complicated programs in Python, as there is most likely a ready-to-use function with the needed algorithm already implemented.

2.5.1 SciPy

According to the official SciPy website "SciPy refers to several related but distinct entities" [37]. The entities in mention are the "ecosystem" of open source software for scientific computing in Python (the SciPy stack), the community of people using and developing the software, conferences dedicated to scientific Python programming and the SciPy library, which is one component of the SciPy stack.

The SciPy stack contains several library components with different core functionalities related to scientific computations in Python. Examples worth mentioning are:

- **NumPy:** A base package implementing N-dimensional array handling
- **SciPy:** A library containing several functions, routines and sub-packages for various scientific computations
- **Matplotlib:** A library with advanced plotting functionality

Scikit-learn

Scikit-learn is a library built on NumPy, SciPy and Matplotlib [38]. The library implements several machine learning methods, as well as functions for regression, data preprocessing, model selection and so on.

CHAPTER

3

FEATURE EXTRACTION

In order to choose features which will perform well in a classification system, it is crucial to have knowledge about what characterizes the different classes which should be distinguished between. To make an informed choice of bowel sound features, it was reasonable to validate that the characteristics of the recorded bowel sounds were in fact coherent with the descriptions found in literature. Validation of the characteristics in the data set provided from APT will be done in section 3.1. The results from this section will then be used further in section 3.2.2 when choosing a set of features.

3.1 Validation of bowel sound characteristics

Summarizing from section 2.1.3, a bowel sound signal seems to be characterized by the following traits:

1. Frequency range of 100 - 500 Hz.
2. Varying amplitude, but generally higher than surrounding noise.
3. Sound durations varying between ca 10 ms and 1 s.

Initial simple tests and transforms were performed on portions of the data set to decide whether these descriptions seemed reasonable.

3.1.1 Frequency spectrum

To get an initial impression of the frequency content of the APT data set, some simple power spectral density (PSD) plots were made over various time intervals of the signal. The PSD of the signal was calculated using the `signal.welch()`-function of the SciPy library [39]. A brief explanation of the method can be found in section 2.2.1.

Figure 3.1 shows the calculated power spectral density plot of the data points measured at an arbitrarily chosen time interval of 10 – 20 minutes after sound recording commenced. Raw data from the time interval is shown in the top plot, PSD plots with linear and logarithmic y-axis, respectively, are shown in the two bottom plots. The PSD was calculated with Welch's method by using `welch` from `scipy.signal` [39]. It is clear that the majority of the signal power lies in the lower frequency band, namely below 100 Hz. This seems reasonable, considering the fact that a large portion of the untreated data is noise, which supposedly should stem from cardiac and respiratory sounds with frequencies below 50 Hz (see section 2.1.3). From the logarithmic plot we see that the PSD curve drops permanently below 1 around 600 Hz, which means that virtually no signal power is contained in the higher frequency bands above 600 Hz. A small peak of power can also be observed around 100 – 200 Hz, which can be seen more clearly in the zoomed in plot in figure 3.2. This might correspond to the bowel sounds, which according to section 2.1.3 should have it's main frequency content in the range 100 – 500 Hz.

To validate the frequency band of bowel sounds, it is necessary to zoom in to a time interval where the bowel sound itself is a more prominent part of the total examined data. This is done in figure 3.3, where the PSD is calculated for a time window where a clear bowel sound is detected (verified by auditory inspection). This frequency spectrum has most of it's power located between 100 and 300 Hz, with less prominent components in the lower bands under 50 Hz.

For comparison, figure 3.4 shows a PSD plot from a time interval of equal length to that in figure 3.3, but from a portion of the signal where no bowel sound can be heard. The frequency spectrum is clearly different from that of the bowel sound. A zoomed version of the plot is found in figure 3.5, displaying a clear majority of power at frequencies below 50 Hz. These findings advocate a simple noise filtering to suppress signal noise below 50 Hz. Nevertheless, it is worth noting the small peaks in the logarithmic plot of figure 3.4 between 150 – 300 Hz. These suggest that the background signals or noise outside of audible bowel sounds might have frequency spectra which overlap with the bowel sound spectrum.

Similar inspections of PSD plots at other time intervals both with and without audible bowel sounds yield similar results. Although this is not a statistically significant way of determining the true frequency band of bowel sounds, it gives valuable insight in what characteristics one might expect from a bowel sound signal. It is also reassuring to see results similar to the ones found in literature.

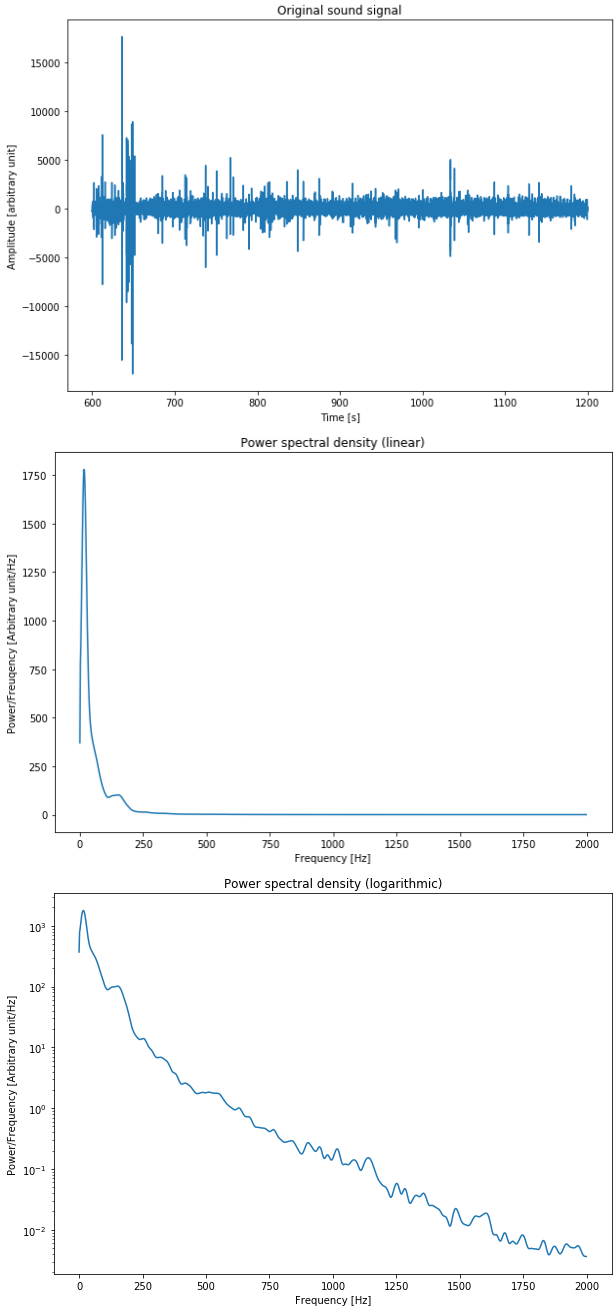

Figure 3.1: Plot of original values (top), linear PSD plot (middle) and logarithmic PSD plot (bottom) of the data from 10 - 20 minutes into the sound recording.

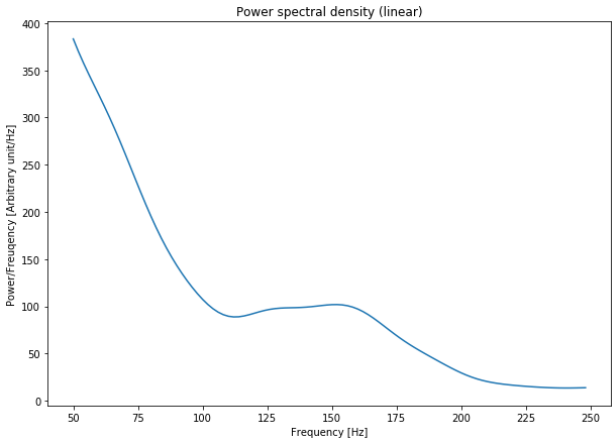

Figure 3.2: PSD plot of the original signal from figure 3.1, zoomed in between 50 – 250 Hz.

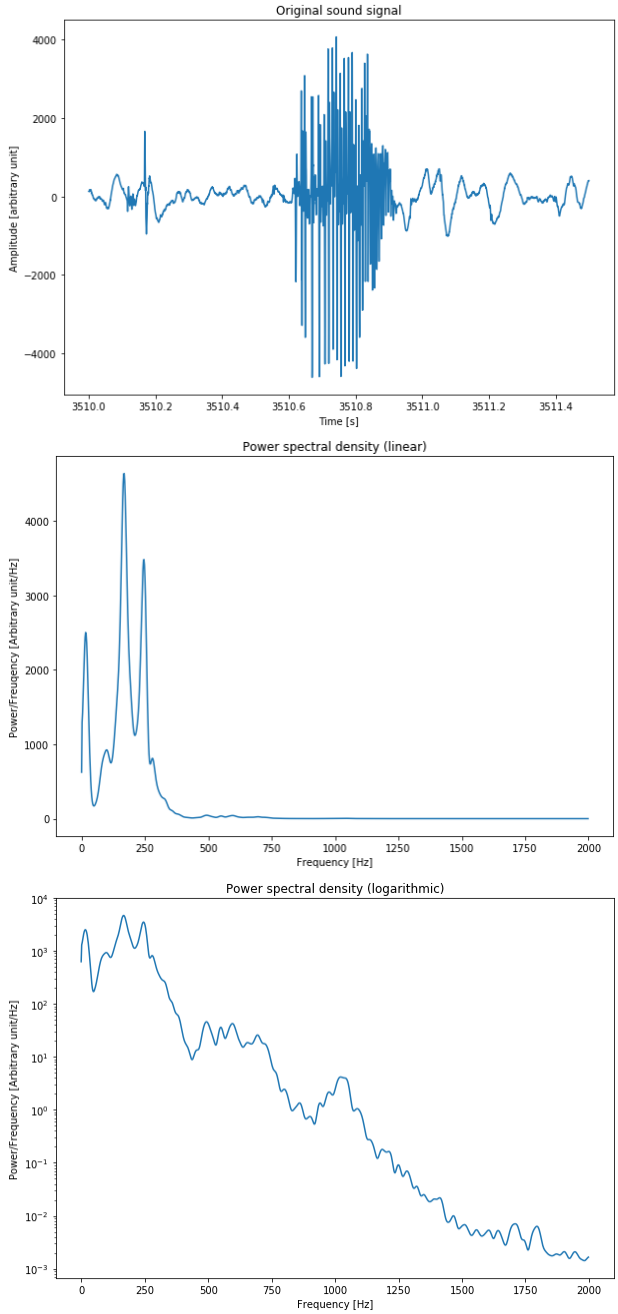

Figure 3.3: Plot of original values (top), linear PSD plot (middle) and logarithmic PSD plot (bottom) of the data at a time interval containing a bowel sound.

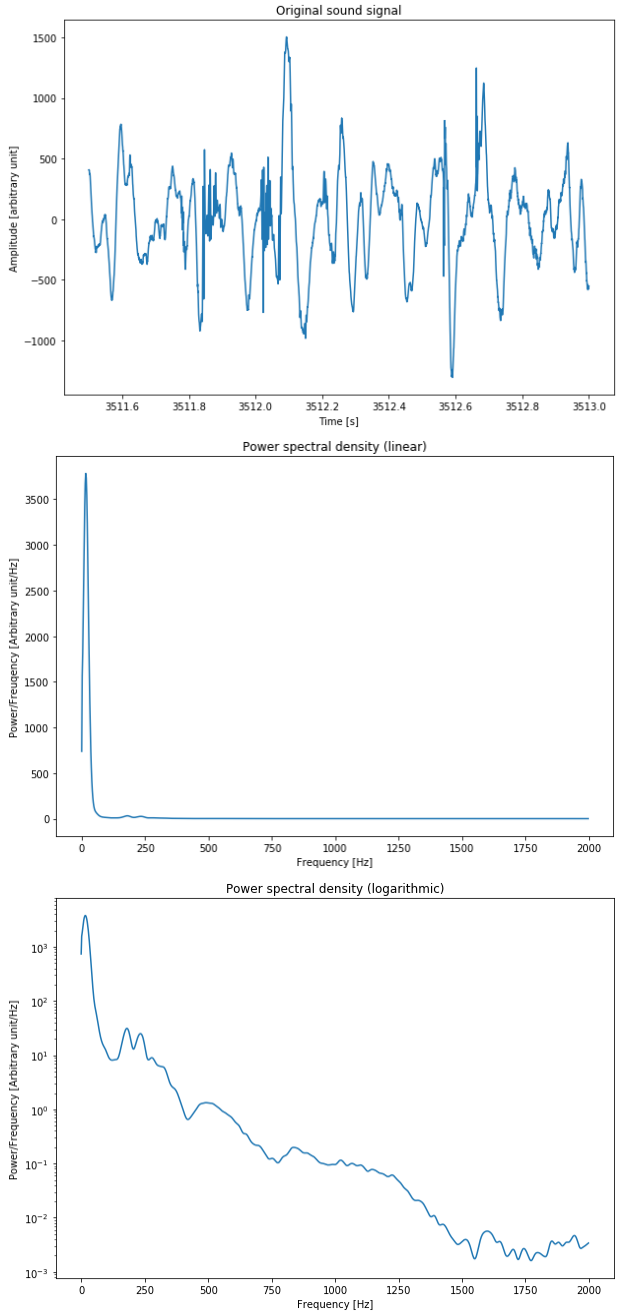

Figure 3.4: Plot of original values (top), linear PSD plot (middle) and logarithmic PSD plot (bottom) of the data at a time interval without any bowel sound.

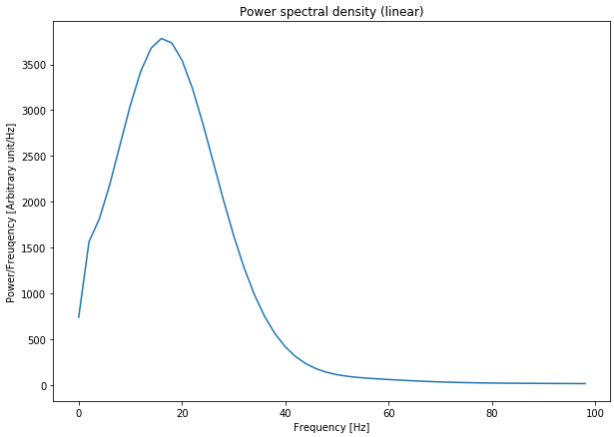

Figure 3.5: PSD plot of the original signal from figure 3.4, zoomed in between 0 – 50 Hz.

3.1.2 Amplitude

The explicit value of the signal amplitude is incredibly varying, just as described in the earlier mentioned references. As an example, figure 3.6 shows two separate bowel sounds, both highpass filtered at 50 Hz to more clearly display the bowel sound in interest. The two sounds have very distinct differences in amplitude, but both can clearly be recognized as bowel sounds by auditory inspection.

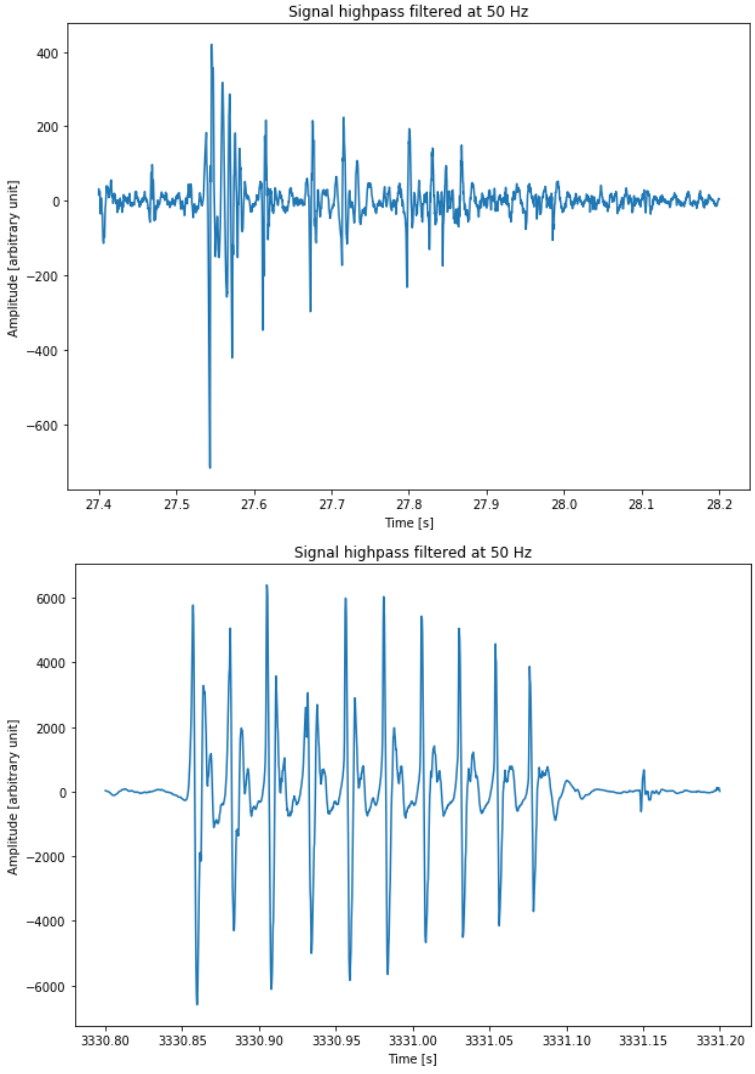

Figure 3.6: Contrast in amplitude between different bowel sounds.

Amplitude values will also depend on the recording equipment used, instrument impedance (as the recorded sound wave is converted to a voltage signal), placement of microphone or stethoscope on the belly and so on. It is therefore meaningless to establish any absolute threshold for a typical bowel sound signal amplitude. Rather, the signal amplitude should be analyzed in the context of the surrounding signal, since a bowel sound almost always features a higher amplitude value than the background signal.

3.1.3 Sound duration

As briefly discussed in section 2.1.3, the academic community has not yet complied any definite description or definition of what a bowel sound actually is. This becomes especially evident when trying to determine the duration of bowel sounds. The plot in figure 2.3 illustrates the ambiguity of defining a typical duration for bowel sounds. When listening to the audio recording, such a waveform sounds like one long, gurgling sound. However, when plotting the sound against a short timescale, it is clear that the sound consists of a series of closely set, short spikes. It is therefore difficult to determine whether or not such a waveform is a cluster of consecutive sounds, or one single sound.

Another issue linked to deciding sound duration is to determine the exact points for sound onset and ending. Figure 3.7 shows a short bowel sound. The spikes with large amplitudes around 497.74 s are clearly the ”main” part of the bowel sound, but weaker ripples are shown both before and after these spikes. Because of the underlying background signals, it is hard to determine exactly where these ripples stop, and which of them are an actual part of the bowel sound. To manually decide this based on the audio clip is also a very difficult task.

These ambiguity issues have been problematic throughout the work with this report. To set a consistent standard for the following work, segments with clusters of consecutive spikes will be treated as one single sound for the rest of this text. Determination of sound onset and ending was done manually after best ability. The resulting range of bowel sound duration found in the examined part of this dataset was approximately 15 ms – 1.7 s.

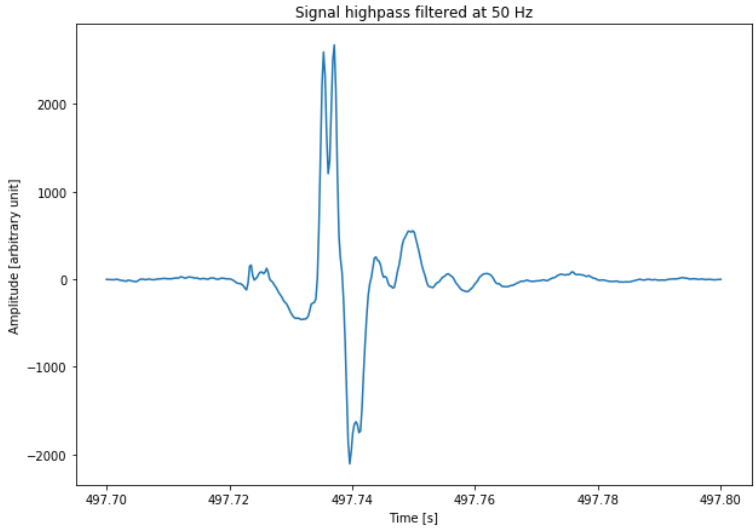

Figure 3.7: Bowel sound illustrating the ambiguity concerning determination of sound onset and ending.

3.2 Extraction of features

3.2.1 Segmentation of data

Keeping in mind the final application of the detection algorithm, namely a real-time detection and quantification system for bowel sounds, it is vital to facilitate real-time classification of a measured sound signal. This means that the detector should more or less continuously decide whether or not the most recent sound samples belong to a bowel sound. However, since frequency information is useful for classification, and complete resolution in both time and frequency is impossible to obtain with traditional integral-based frequency analysis (see section 2.2.1), a certain portion of the signal must be considered for each feature calculation. A compromise between these two needs is to implement a sliding window approach. This means that a window of a fixed number of samples are evaluated each timestep, and the computed values stored into a feature vector such as in equation 3.1.

$$\mathbf{x}_{k+1} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix} \tag{3.1}$$

For each timestep k, the feature vector $\mathbf{x}_k$ consists of d features $x_i,\ i = 1, ..., d$. Each time a vector has been computed, the window slides one sample forward and the process is repeated. The segmentation process is illustrated in figure 3.8.

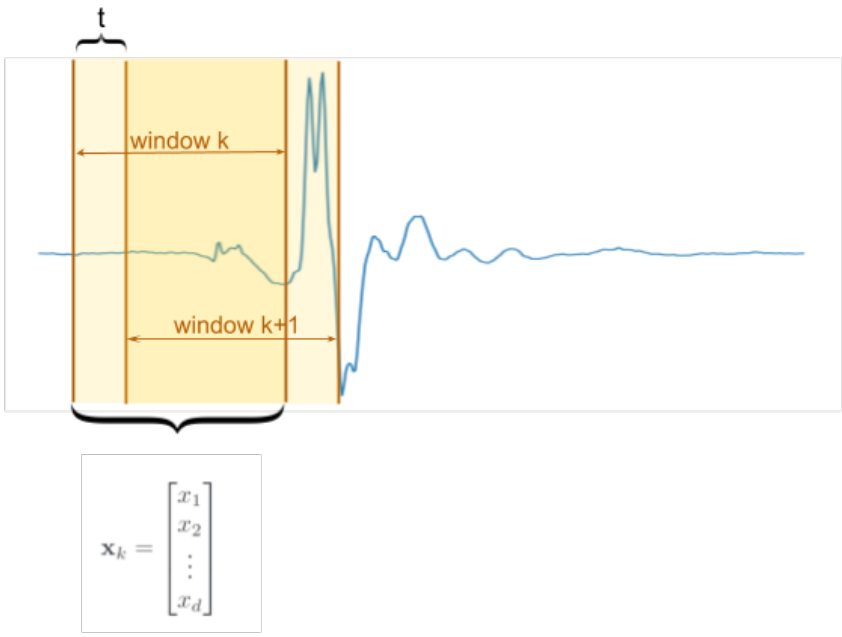

Figure 3.8: Illustration of sliding window segmentation.

It is important to ensure that the window size is appropriate to capture the wanted information. Figure 3.9 illustrates the considerations required to keep in mind when choosing an appropriate width for the feature evaluation window. The figure shows a 100 Hz sinusoidal signal and a very short bowel sound with a duration of approximately 15 ms.

The first point to consider when choosing a time window is explained nicely by [23, p. 60]: ”Since frequency is directly proportional to the number of cycles per unit time, it takes a narrow time-window to locate highfrequency phenomena more precisely and a wide time-window to analyze lowfrequency behaviors more thoroughly.” As explained in section 3.1.1 we are interested in the frequency content between 100 – 500 Hz. This corresponds to a signal period of maximum $T = \frac{1}{100\,\mathrm{Hz}} = 0.01\,\mathrm{s} = 10\,\mathrm{ms}$. Thus, the window size must be at least 10 ms in order to detect the lowest bowel sound frequencies, as illustrated by the minimal limits in figure 3.9. However, if the window is too wide, a short bowel sound will only make up a small part of the analyzed samples. This would mean that the calculated features of the window would contain a mixture of bowel sound-characteristic values and values corresponding to background signals. This would make it difficult for the classifier to recognize that the window in fact contains a bowel sound. Section 3.1.3 explained that minimal bowel sound duration is approximately 15 ms. Considering a time instant such as the one in figure 3.9, where the window (limited by blue lines) is positioned such that the middle of the window aligns with the middle of a 15 ms wide bowel sound, we would want the window to be so small that the majority of the window is filled by the bowel sound. This yields a maximum window size of about 20 – 30 ms.

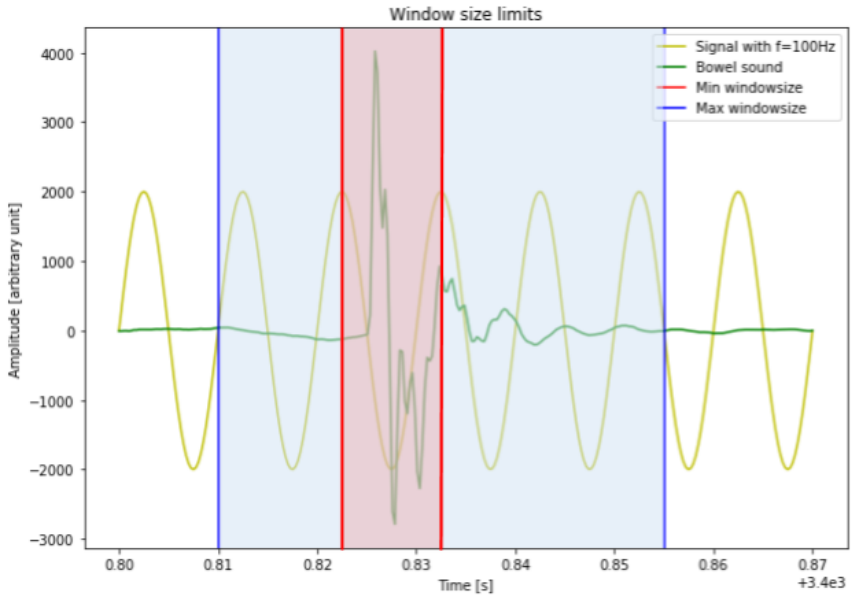

Figure 3.9: Illustration of minimal and maximal widths of the segmentation window.

3.2.2 Choice of features and signal processing methods

The first processing done on the signal was simple noise filtering. Section 3.1.1 found that the background signals had the majority of their frequency content below 50 Hz. A 50 Hz highpass filter was therefore applied to the data before any further calculations were performed. A Butterworth filter of 3rd order was chosen because of the maximally flat passband response described in section 2.2.2. The code implementation of the filter consisted of a combination of the functions `butter` and `filtfilt`, both from the `signal`-package of the SciPy library [39]. `butter` returns the filter coefficients of a Butterworth filter with order specified by the user. `filtfilt` applies a given filter twice to an input signal; once forwards and once backwards. This remedies the phase distortion of the Butterworth filter and yields an output signal with no phase shift.

Based on the bowel sound characteristics presented in section 2.1 it seems reasonable to choose features which enable the classifier to detect an increase in power within a certain frequency band, as well as changes in signal amplitude. Table 3.1 displays the features chosen as candidates for the classifier. These are not necessarily the features which will give the best classification performance, but rather a starting point for selection of the optimal feature combination.

Features 1 and 2 are ways to represent the increased volume or loudness of the sound signal when encountering a bowel sound. Feature 2, mean nonlinear energy, was used as a feature for classification of electroencephalogram (EEG) signals by [40], as one of the top performing features. Since classification of EEG and bowel sound signals share

Feature number	Feature name
1	Mean absolute amplitude
2	Mean nonlinear energy (MNLE)
3	Mean PSD around 400 Hz
4	Mean PSD around 300 Hz
5	Mean PSD around 200 Hz
6	Ratio of BS amplitude/BG amplitude
7	Ratio of BS MNLE/BG MNLE
8	Ratio of BS PSD around 400 Hz/BG PSD around 400 Hz
9	Ratio of BS PSD around 300 Hz/BG PSD around 300 Hz
10	Ratio of BS PSD around 200 Hz/BG PSD around 200 Hz
11	Signal envelope

Table 3.1: Table of features.

some common traits, this feature was thought relevant for this application. Feature 3 – 5 reflects the findings from section 2.1.3 about the frequency content of bowel sounds. The power spectral density was chosen as a representation of signal frequency content. The wavelet transform was deselected as a frequency representation, as the great advantage of this method – time resolution of the frequency content – is not of any great importance within a short time window. Features 6 – 10 are the ratios of the previous features within the current window, with respect to the same features calculated in a window already classified as background signal (no bowel sound). The ratio features should ideally correct for the large variety of signal characteristics, both between bowel sounds and at different locations of the background signal. Lastly, feature 11 is the signal upper envelope, which is an outline of the bowel sound shape. An example of the envelope around a bowel sound can be seen in figure 3.10. This feature traces the macroscopic changes of the sound wave, and ignores the very rapid changes and transients typical for the bowel sounds. A signal envelope curve is created by performing a Hilbert transform on the signal, and then plotting the absolute value of the transformed data. See section 2.2.3 for a brief description of the Hilbert transform.

3.2.3 Calculation of features

The following paragraph will describe the calculation of features for each window. The original discrete sound signal is denoted $x[n]$, where $n \in [0, ..., N-1]$ is the sample number and N is the total number of samples in the signal. Each window consists of the samples $x[n], ..., x[n+k]$ where the number of samples k in each window is equal for all windows. A window size of 30 ms was chosen, which gives $k = 120$ samples in each window. The window size was set as large as possible within the guidelines of section 3.2.1. This because a higher number of input samples to the PSD calculations gives a bet-

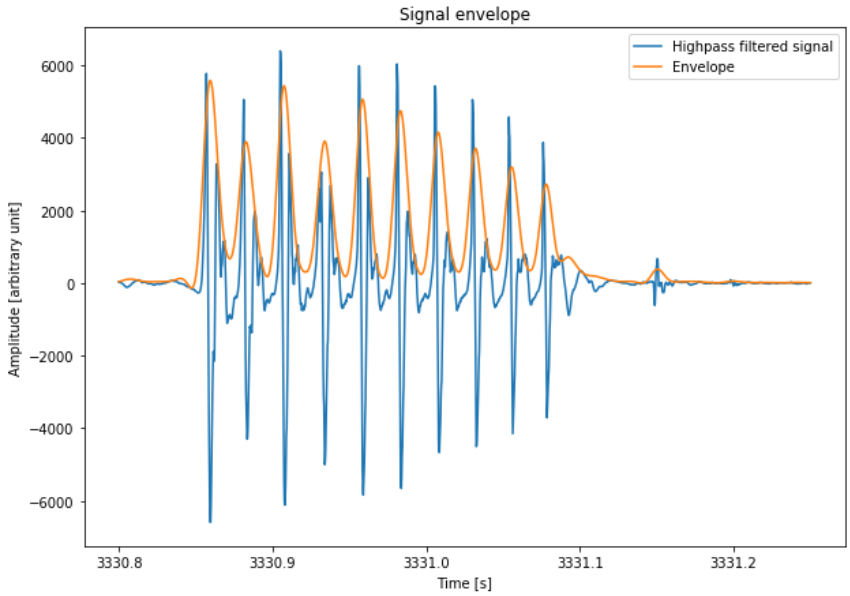

Figure 3.10: The lowpass filtered envelope of a bowel sound.

ter frequency resolution on the output.

Mean absolute value

Feature number 1 is simply the mean of the absolute value of all the data points within the current window. It is calculated with equation 3.2.

$$|x|_{mean} = \frac{\sum_{i=n}^{n+k} |x[i]|}{k} \tag{3.2}$$

Mean nonlinear energy

Feature number 2 is calculated by equation 3.3.

$$\text{MNLE} = \frac{\sum_{i=n}^{n+k} \left(x[i]^2 - x[i-1]x[i+1]\right)}{k} \tag{3.3}$$

Mean power spectral density

Features 3 – 5 are the window's power spectral density in the frequency bands around 400 Hz, 300 Hz and 200 Hz, respectively. A bandwidth of 100 Hz was chosen, such that

feature 3 is the mean of the window's PSD between 300 – 500 Hz, feature 2 is the mean PSD between 200 – 400 Hz and feature 3 is the mean PSD between 300 – 500 Hz. The PSD was calculated with Welch's method (see section 2.2.1), and implemented in code by using the `welch` function of the `scipy.signal` package [39].

Ratio features

Features 6 – 10 are the ratios of feature 1 – 5 calculated for the current window, with respect to the same features calculated for windows with background (BG) signals. The detection algorithm will continuously save the feature vectors for the last 5 windows containing background signals. For each of the features 1 – 5 in the current window (CW), the feature value from the last 5 BG vectors will be averaged, and the corresponding ratio value will be computed with equation 3.4.

$$\text{FEATRATIO}^{CW} = \frac{\text{FEATVAL}^{CW}}{\frac{1}{5}\sum_{j=0}^{5}\text{FEATVAL}_j^{BG}} \tag{3.4}$$

Signal envelope

The Hilbert transform of each window was calculated with the function `hilbert` from `scipy.signal` [39]. This yields the analytic representation of the signal, where the absolute value of this is the signal envelope. In addition, a 4th order Butterworth lowpass filter with 70 Hz cutoff (`butter` and `filtfilt` was utilized for this) was applied to the envelope signal to produce the curve displayed in 3.10. This was done to suppress the most rapidly varying components of the signal, and let the envelope smoothly trace the prominent shape of the bowel sound. To increase computational efficiency and reduce memory usage, the signal envelope signal was downsampled such that only every 10th sample was included in the final features.

CHAPTER

4

RECOGNIZING BOWEL SOUNDS

4.1 An overview of the classification system

To give an overview of the classification system used to detect bowel sounds, figure 2.8 in section 2.3.1 will be used as a reference. The sensor module takes care of data acquisition and representing the signal's physical properties as numerical raw data. This process is further explained in section 2.1.2. The array of raw data is segmented by use of a sliding window, and features are calculated for each window as explained in section 3.2.1. This process corresponds to the feature extraction module in figure 2.8. Development of the classifier module will be further explained in this chapter. This development process is generally described in section 2.3.1, and illustrated by the upper part (*training phase*) of figure 2.9. To construct an effective classifier, both a training data set and a training strategy is needed.

After a finished classifier module is constructed and trained, detection of bowel sounds in a new and unseen sound file will happen in real time. The measured sound waves will be continuously converted to numerical raw data values, and the sliding window will move along the newly acquired data. Feature vectors will be calculated for each timestep just as in chapter 3. Each calculated feature vector can then be sent into the classifier, just as in the *usage phase* of figure 2.9, and the classifier will predict whether or not this particular window (or at least major part of the window) contains a bowel sound.

4.2 Construction and splitting of data set

A raw sound signal was delivered as a basis for the data set, such as explained in section 2.1.2. A portion of the original sound file was selected as a data set, since processing the entire sound clip would require excessive processing power. Section 4.4 will explain how signal portions were extracted from the original sound file to create the total data set. In total, 92.4 seconds of sound was used to construct a data set. With a sampling rate of 4000 Hz, this gives $92.4\,\mathrm{s} \times 4000\,\mathrm{Hz} = 369\,600$ samples. After the feature extraction process from section 3 we are left with a set of feature vectors which make up the points of out total data set.

As explained in section 2.3.4, it is important to separate the data set into a training set, a test set and a validation set. The total data set was randomly split into test and training sets by use of the function `model_selection.train_test_split` from the `sklearn`-library [41]. Validation is done in "real time" (timestep by timestep) on a part of the sound file not included in the original total data set, so as to simulate how the classifier will work when used as intended. Note that processing time or performance will not be considered as part of the validation, as explained in section 1.2.2.

4.3 Choice of pattern recognition method

To express the bowel detection problem in machine learning terms; the task at hand is to separate the calculated feature vectors (which each correspond to a sample of the sound signal) into two discrete categories; "bowel sound" or "no bowel sound". Our problem is thus a classification problem. The solution strategy of this report has been to carefully select features which are thought to separate the two classes fairly well. This will hopefully make it possible to obtain decent results event with a fairly simple classification algorithm.

It was explained in section 2.1.3 that there is little common knowledge about the characteristics of bowel sounds. This also means that it is difficult to have any knowledge about how the different features from section 3.2.2 are distributed for bowel sounds in general. This lack of information suggests that nonparametric methods are best suited for this use.

4.3.1 Test of unsupervised algorithm

Marking the data set is a time consuming activity, and lack of medical expertise might also lead to errors in labeling. These considerations encourage an initial test of unsupervised training approaches. The Scikit-learn User Guide features a handy cheat sheet [42] to help with picking an initial algorithm when starting to develop a machine learning application. The bowel sound detection application consists of predicting a category for each sample, where the number of categories are known. In addition, the amount of training samples is fairly large. According to [42], the KMeans clustering algorithm is a good pick for initial testing. An explanation of clustering and KMeans can be found in section 2.3.2. `cluster.KMeans` from `sklearn` was used to cluster the dataset.

After splitting the data set as explained in 4.2, the training set was used to fit the k-means model and compute the two centroids $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2$. After this, new samples from the validation set could be classified by assigning them to their nearest centroid in feature space. A quick overlook of the results did however yield unpromising results. As we see in figure 4.1, all of the samples in the two time intervals have been sorted into the same cluster, even though there are clearly both bowel sounds and background signal present.

A plot containing a part of the signal basis for the training set is found in figure 4.2 together with their corresponding cluster labels. Only a small portion of samples have been assigned to cluster number 1, while the majority is sorted into cluster 0. The division is seemingly completely uncorrelated to whether or not the sample is part of a bowel sound.

The terrible clustering results could be explained by a number of reasons. The data set could for instance have a lot of outliers tampering with the cluster means or the shape of the data set in feature space could be suboptimal for this particular clustering algorithm. Spending time on costumizing the data set and searching for a better clustering algorithm might not be completely productive, as clustering algorithms often fail to provide satisfactory classification results. Rather, a supervised approach will be tested on our data.

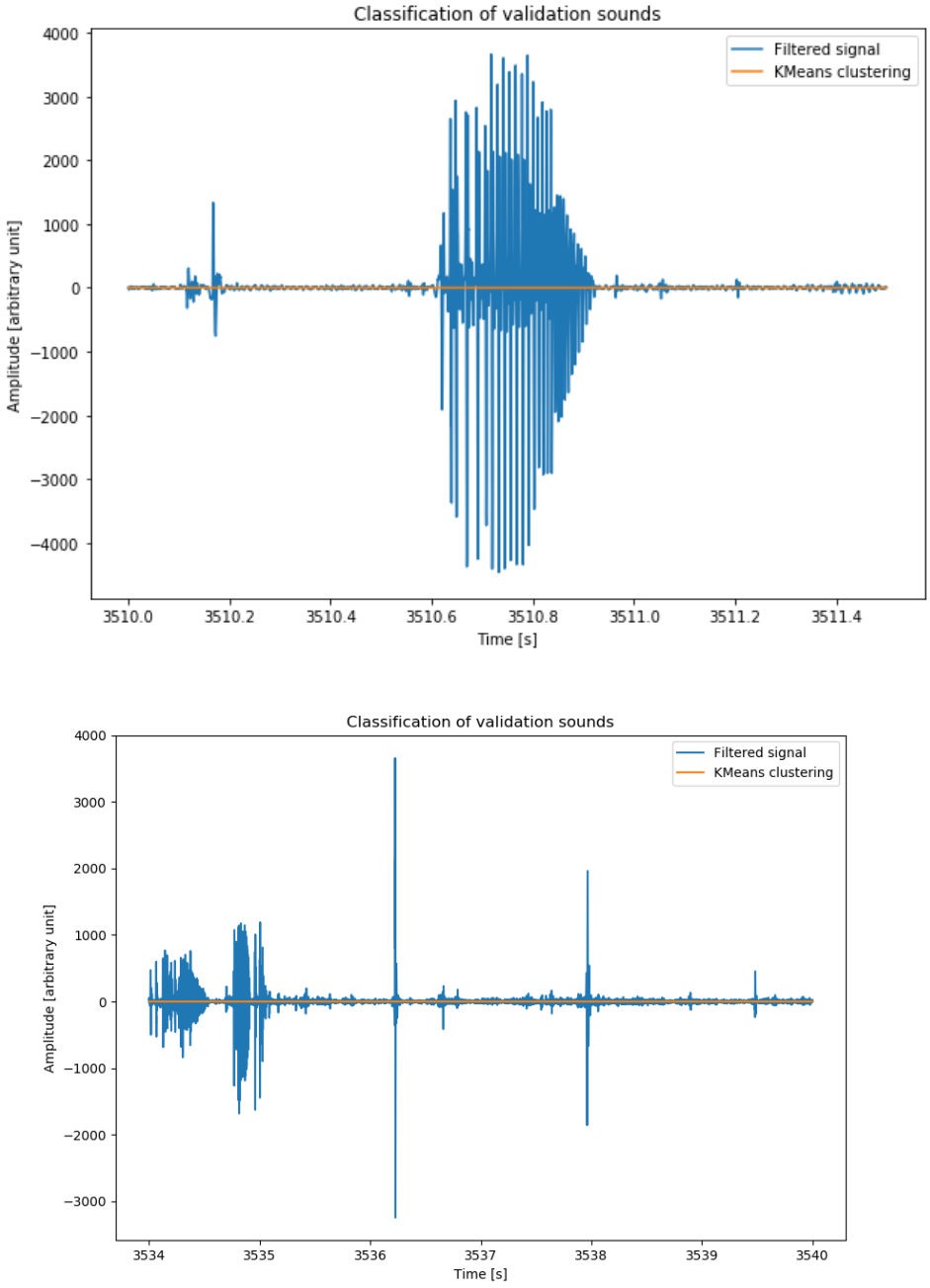

Figure 4.1: Class predictions by `KMeans` for a portion of the validation samples. The orange line is the integer value of the class label (cluster number) assigned to each sample. In both plots, all samples belong to the same cluster.

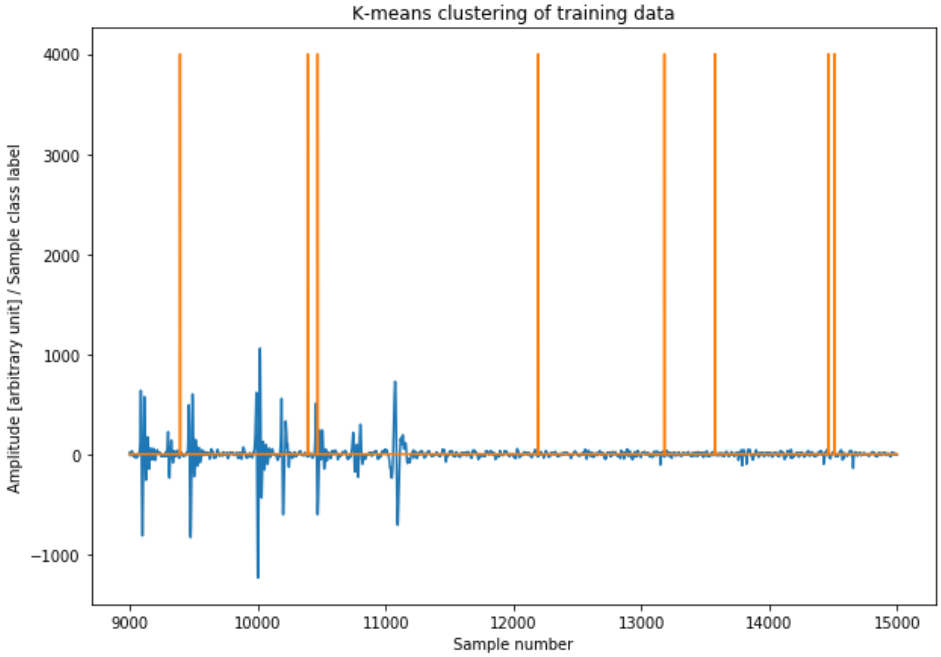

Figure 4.2: K-means clustering results for training data plotted together with the filtered signal for the corresponding samples. Cluster (class) label values have been multiplied by 4000 for visibility.

4.3.2 Supervised algorithm

Section 4.3.1 shows that clustering yields unsatisfactory results for the detector. Another approach is to use a supervised algorithm, which needs labeled data. Description of how data labels were generated is found in section 4.4.

As mentioned in the beginning of section 4.3, features for the classifier were chosen in a way that hopefully separates the two classes well. It is however hard to visually verify that the data is in fact linearly separable (or at least mostly separable) when the dimension of the feature vector is as high as in this case. As explained in section 2.3.3, a data set that is linearly separable in higher dimensions is not necessarily still linearly separable when projected onto a lower dimensional space. However, if the data can be classified well by a linear SVM, the set is linearly separable (at least to a sufficient degree). The next choice of classification algorithm is therefore an SVM classifier. Soft margin SVM is chosen over hard margin, since it is unknown how well the classes are separated in feature space. `linear_model.SGDClassifier` from `sklearn` implements a linear classifier which is trained by using a stochastic gradient descent (SGD) algorithm to minimize a chosen loss function [41]. The loss function is chosen such that a minimal solution yields an optimal discriminant function for the classifier, given the training set at hand. Different choices of loss function results in different classifier algorithms. When choosing a Hinge loss function for `SGDClassifier`, a linear soft margin SVM is the result. Performance and results of `SGDClassifier` on the bowel sound data set can be found in chapter 5.

4.4 Marking the data set

The data set was marked manually by auditory inspection. To determine the onset and ending of each sound, the highpass filtered signal was inspected visually. No experts were involved in marking the dataset, so the data might contain mislabeled samples. Nevertheless, marking was done after best effort by excluding all ambiguous signal portions from the set. As mentioned in section 3.1.3, sounds which could be characterized as a cluster of short sounds have rather been treated as longer, coherent bowel sounds during labelling. The final marked data set contains explicitly marked samples of either ”bowel sound” or ”no bowel sound”.

Since the sound signal is marked sample by sample, and feature vectors are calculated for a collection of samples (the window), it is necessary to match each sample label to a corresponding feature vector (a corresponding window). The middle sample of a window was chosen as the current target label for each window. This is illustrated in figure 4.3, where a window contains samples labeled both as ”bowel sound” and ”no bowel sound”. As the middle sample has been classified as ”bowel sound”, the whole window is then considered to contain a bowel sound, and the calculated feature vector will be paired with a ”bowel sound”-label.

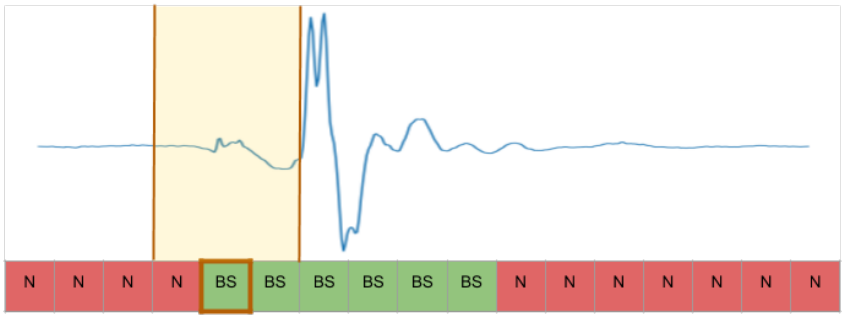

Figure 4.3: Matching of the sample labels to corresponding segmentation windows. The array illustrates sample labels. Size of samples, window and bowel sound are not to scale. The label of the middle sample of the window is chosen as the feature vectors corresponding label.

4.5 Feature selection

Since there were no prior knowledge about the optimal number of features, a recursive feature elimination method with cross-validation was chosen. Scikit-learn has implemented such a method in the class `feature_selection.RFECV`. The class constructor takes a classifier (or other estimator) as a parameter, on which to perform the cross-validation. In addition, the labeled training data has to be provided. Note that only the training set returned from `model_selection.train_test_split` was used as an input, and that the `RFECV`-function performs further splitting of this data set in the cross-validation procedure. Thus, an ”untouched” test set is left to calculate the final metrics of the classifier after fitting it with the final feature set. There are also a number of other parameters to tune, such as the number of features to eliminate per iteration and which scoring function to use as a performance measurement during cross-validation.

The number of features to eliminate per iteration was set to 1. As a scoring function, the mean accuracy (see equation 2.16) of the `SGDClassifier` was used. The initial training set provided to the algorithm contained all the original features listed in table 3.1. Figure 4.4 shows how the classifier accuracy develops as a function of the number of features used for training. The graph has a peak at 5 features, indicating that the optimal feature subset contains 5 features.

The RFECV algorithm chose the features shown in table 4.1 for the optimized feature set. The signal envelope features are apparently not performing well, and additionally require quite a lot of processing time to compute due to the Hilbert transform. It was therefore decided to cut out the signal envelope entirely from the feature set, even though the last sample was chosen by the RFECV-algorithm. The feature selection scheme was therefore

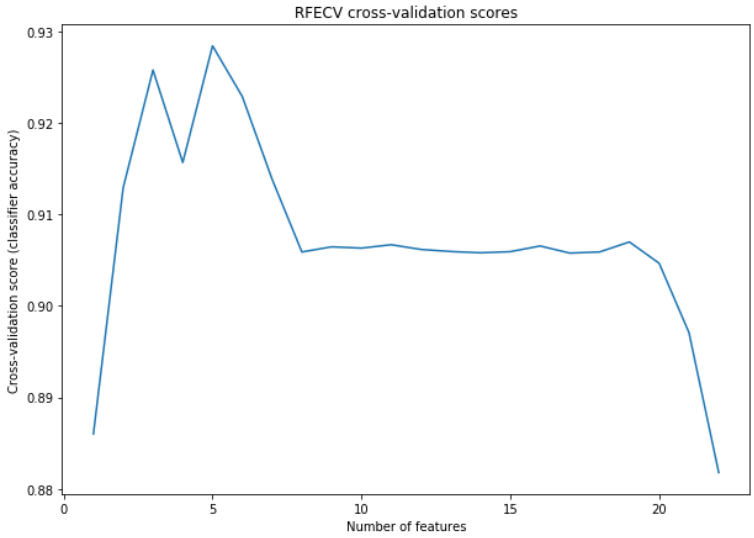

Figure 4.4: Cross-validation scores from the RFECV algorithm.

Feature number	Feature name
3	Mean PSD around 400 Hz
6	Ratio of BS amplitude/BG amplitude
7	Ratio of BS MNLE/BG MNLE
10	Ratio of BS PSD around 200 Hz/BG PSD around 200 Hz
11	Last sample of the signal envelope

Table 4.1: Features chosen by RFECV in the initial execution.

executed once more, this time with features 1 through 10 from table 3.1 included in the training set provided to the RFECV-function.

The resulting cross-validation score graph is displayed in figure 4.5. This time, a set of 6 features yielded the highest classifier accuracy in the cross-validation tests. The final set of chosen features is listed in table 4.2.

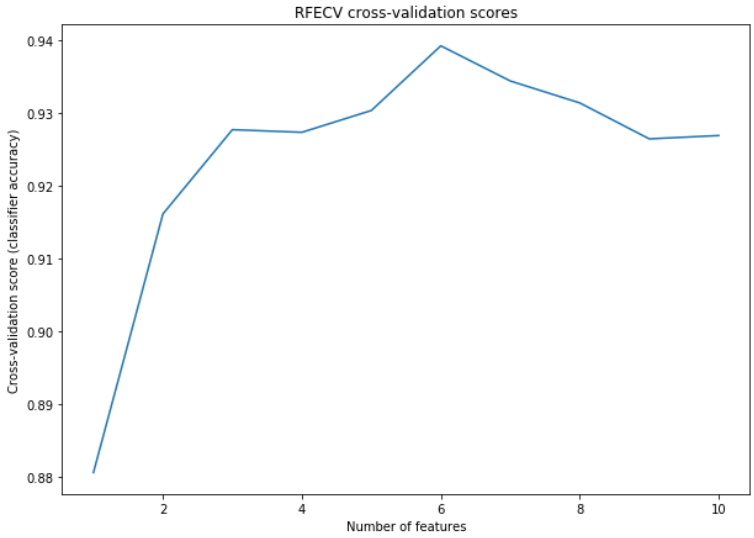

Figure 4.5: Cross-validation scores from the RFECV algorithm.

Feature number	Feature name
1	Mean absolute amplitude
3	Mean PSD around 400 Hz
6	Ratio of BS amplitude/BG amplitude
7	Ratio of BS MNLE/BG MNLE
9	Ratio of BS PSD around 300 Hz/BG PSD around 200 Hz
10	Ratio of BS PSD around 200 Hz/BG PSD around 200 Hz

Table 4.2: Final features chosen by RFECV in the second execution.

CHAPTER

5

RESULTS

With the features listed in table 3.1 of section 3.2.2 and `SGDClassifier(loss="hinge")` as explained in 4.3.2, the results presented below were obtained. The number of iterations for the SGD algorithm was found through trial and error to be 3000 iterations. This gave a nice trade-off between processing time and classification accuracy.

5.1 Classification without feature selection

The classifier was first trained on the original feature set listed in table 3.1. Metrics obtained from classification of the test set, as well as plots of the class predictions done for the validation samples follow below.

5.1.1 Classifier metrics

To evaluate the classifier, a labeled test set was used to calculate the classifier's confusion matrix (see section 2.3.4). For a test set with $n = 92402$ samples, the confusion matrix in table 5.1 was obtained. From table 5.1 we can calculate the classifier metrics presented in table 5.2. For details on how each metric is calculated, please refer to section 2.3.4.

A simple and intuitive metric to examine first is the mean accuracy of the classifier, or the rate of correctly classified samples (regardless of label value). This metric has a quite high value, showing that the developed classification algorithm supposedly works well overall. The error rate is the complimentary value of the accuracy, thus having a quite

		Predicted class	
		NS	**BS**
True class	**NS**	67 937	7 516
	BS	1 815	15 134

Table 5.1: Confusion matrix calculated from evaluation of the classifier without feature selection. NS signifies ”no bowel sound” and BS ”bowel sound”.

Metric	**Value**
Mean accuracy	0.90
Error rate	0.10
Sensitivity	0.89
Specificity	0.90
Precision	0.66

Table 5.2: Evaluation metrics for the SVM classifier without feature selection.

low value. However, these metrics do not give a complete picture of the classifier's performance. They are both dependent on the *prevalence* of bowel sounds in the data set, or the ratio of actual bowel sound samples to background signal samples. In the provided data set, there is a significantly higher amount of background signal samples than of bowel sound samples. Of the total $9\,240$ samples, $16\,949$ are bowel sound samples and $75\,453$ are background signal samples. A classifier categorizing all samples as ”no bowel sound” would therefore achieve a fairly good accuracy score, but obviously be a terrible classifier. Other metrics must therefore be observed in order to establish the true performance of the implemented algorithm.

Sensitivity measures the classifier's ability to find all bowel sounds which are present. An algorithm with 100 % sensitivity would not overlook a single bowel sound. However, it might also classify a lot of the background signal as ”bowel sounds”. Specificity tells us how good the classifier is at rejecting the background signal. A classifier with 100 % specificity would classify every single background signal sample as ”no bowel sound”, but may detect very few of the present bowel sounds. Table 5.2 displays quite high levels for both sensitivity and specificity, yielding a good balance between detecting present bowel sounds, but also rejecting background signal.

Another metric quite popular in evaluation of machine learning applications is the precision. This value reflects how many of the samples classified as ”bowel sound” by the classifier are actually bowel sounds. In this way it measures the relevance of the classifier results, whether or not a ”bowel sound”-classified sample can be trusted to actually be a bowel sound. The precision is however also dependent on prevalence, and will therefore have a lower value because of the overall low bowel sound occurrence rate. The metric is

still included in order to compare it to the classifier trained with feature selection.

5.1.2 Visual evaluation of validation set

The following figures contain the classification results for selected intervals of the validation set. The high pass filtered signal which is the basis for feature calculation is plotted together with the class label predicted for each sample.

The classifier quite accurately classifies the bowel sound samples, both for the larger, more prominent sounds, shorter sounds and sounds with lower amplitude. Especially in figures 5.1, 5.2 and 5.3 the sounds are identified fairly precisely, even though the plots feature sounds of different duration, shape and amplitude.

Onset and ending of sounds are in some places slightly shifted compared to how the data set in question would have been marked. This is especially visible in figure 5.4. In addition, in figures 5.4 and 5.5, the classifier has split up what might be considered as one continuous sound into several consecutive sounds. This issue of determining duration of a sound was discussed in section 3.1.3.

It is also worth noticing that the classifier sometimes will detect a bowel sound in samples with very low signal amplitude, which would probably have been labeled as ”no bowel sound” by manual marking of data. An example is the short interval of ”bowel sound”-classified samples around 3516.4 seconds in figure 5.3. This could be an erroneous classification from the classifier, or it could be an actual bowel sound that is too faint to be heard by auditory inspection.

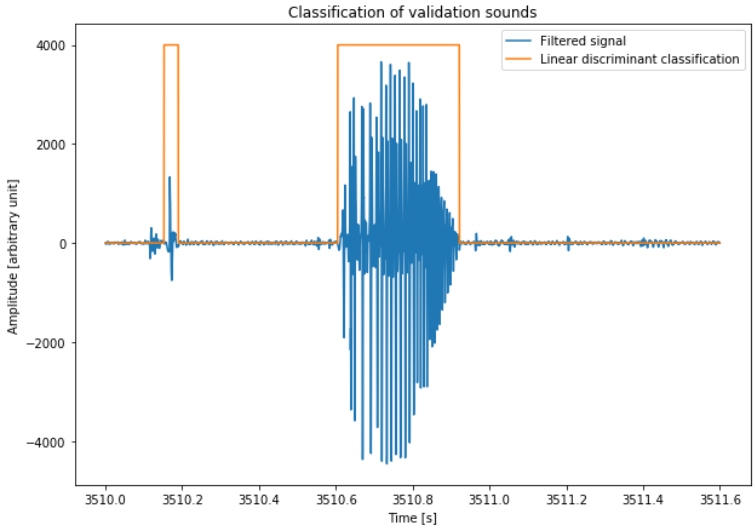

Figure 5.1: Prediction results for the validation set. Class label value has been multiplied by 4000 for visibility.

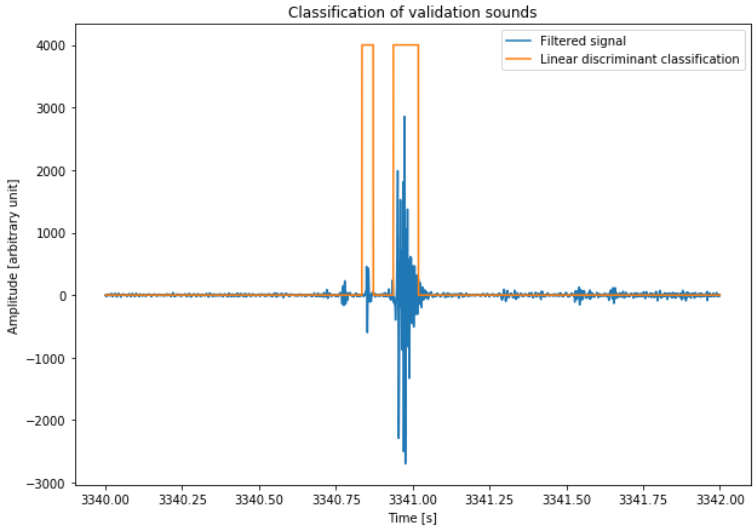

Figure 5.2: Prediction results for the validation set. Class label value has been multiplied by 4000 for visibility.

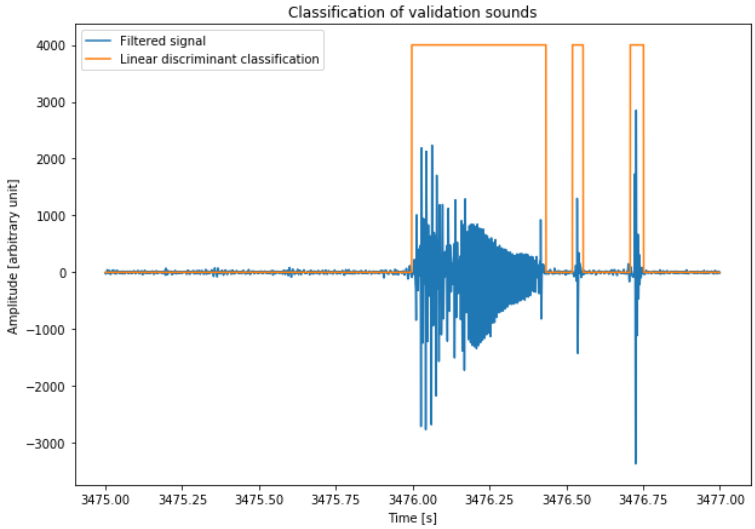

Figure 5.3: Prediction results for the validation set. Class label value has been multiplied by 4000 for visibility.

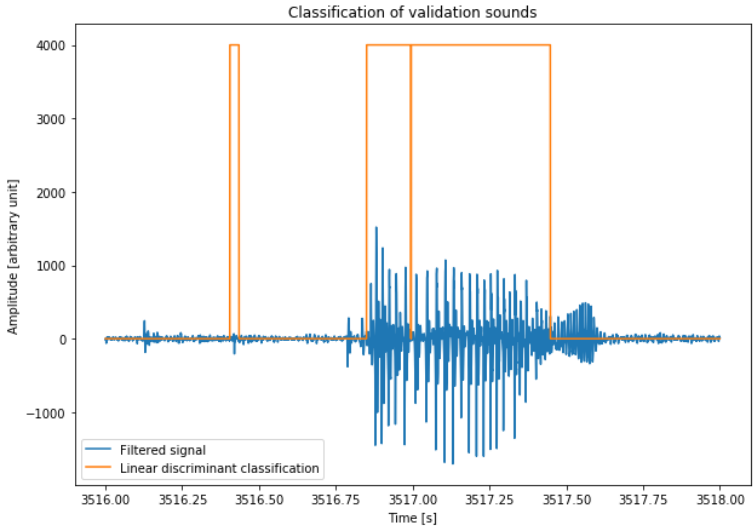

Figure 5.4: Prediction results for the validation set. Class label value has been multiplied by 4000 for visibility.

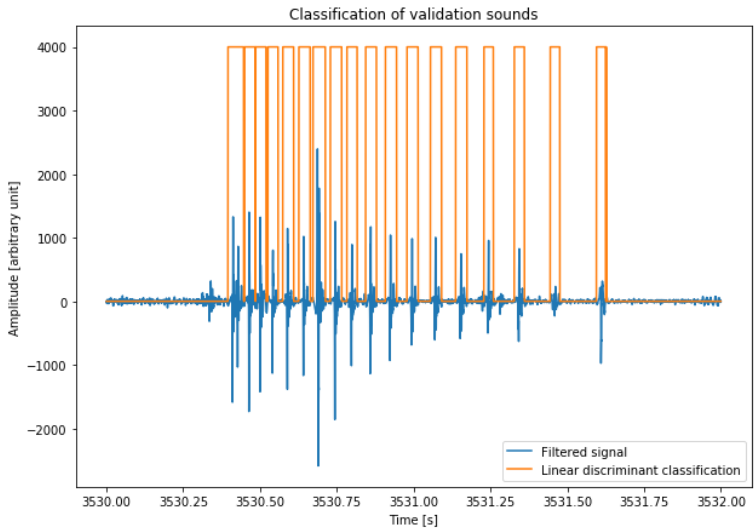

Figure 5.5: Prediction results for the validation set. Class label value has been multiplied by 4000 for visibility.

5.2 Classification with feature selection

After feature selection was performed as explained in section 4.5, the `SGDClassifier` was trained again. This time, the optimal subset of features listed in table 4.2 was used as an input, while all other parameters remained unchanged.

5.2.1 Classifier metrics

A confusion matrix was obtained for the newly trained classifier in the same way as described in section 5.1.1. The matrix is shown in table 5.3. Again, metrics were computed from the obtained confusion matrix. These results are displayed in table 5.4.

		Predicted class	
		NS	**BS**
True class	**NS**	69 156	4 939
	BS	596	17 711

Table 5.3: Confusion matrix calculated from evaluation of the classifier with feature selection. NS signifies ”no bowel sound” and BS ”bowel sound”.

Metric	**Value**
Mean accuracy	0.94
Error rate	0.06
Sensitivity	0.97
Specificity	0.93
Precision	0.78

Table 5.4: Evaluation metrics for the SVM classifier with feature selection.

The metrics indicate that the classifier performance has increased compared to the classification algorithm without feature selection. All values in table 5.4 have increased compared to table 5.2, except the error rate, for which a decrease was desirable. It is worth noting the shift that has happened in sensitivity and specificity. In table 5.2, the sensitivity is lower than the specificity, indicating that the classifier in section 5.1 is better at predicting background signal than bowel sound signals. The opposite is true for this classifier, even though both predictive values have improved compared to section 5.1.1. The precision value has also gone up for this classifier. This implies that the bowel sounds detected by the classification algorithm with feature selection will be more ”trustworthy” than the ones detected by the classifier in section 5.1.

5.2.2 Visual evaluation of validation set

Classification of the validation set was done in the same manner as in section 5.1.2. For comparison, results are plotted for the same time intervals as in section 5.1.2.

The overall tendency for all plots is that this classifier is more liberal with labelling samples as ”bowel sound”, compared to the previous classifier. This can be linked to the increased sensitivity, since this means that the algorithm is less likely to ”overlook” present bowel sounds. This increased rate of bowel sound labelling has led to less fragmentation of the sounds in figures 5.9 and 5.10, compared to figures 5.4 and 5.5. It has also remedied the problems concerning detection of sound onset and ending. This is especially visible when comparing figure 5.4 with figure 5.9. However, the increased sensitivity effect also leads to a lot of very small peaks being classified as bowel sounds. These are not audible (at least for an untrained ear) in the sound file, and whether or not these sound signals stem from the intestines is difficult to determine without expert analysis.

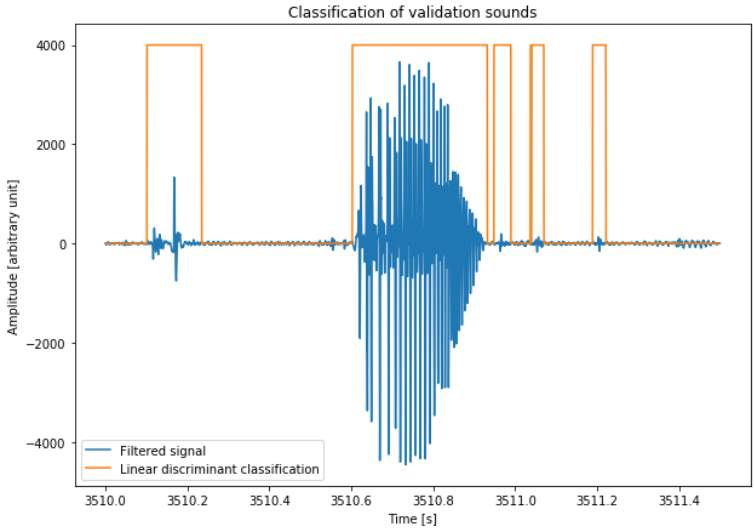

Figure 5.6: Prediction results for the validation set. Class label value has been multiplied by 4000 for visibility.

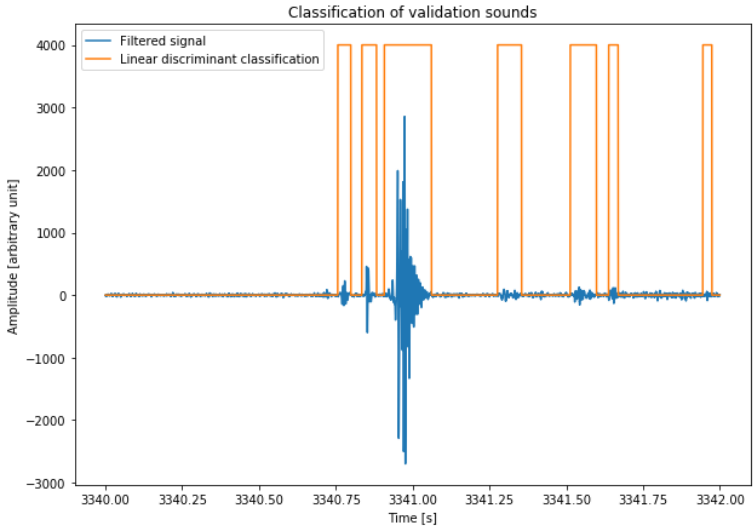

Figure 5.7: Prediction results for the validation set. Class label value has been multiplied by 4000 for visibility.

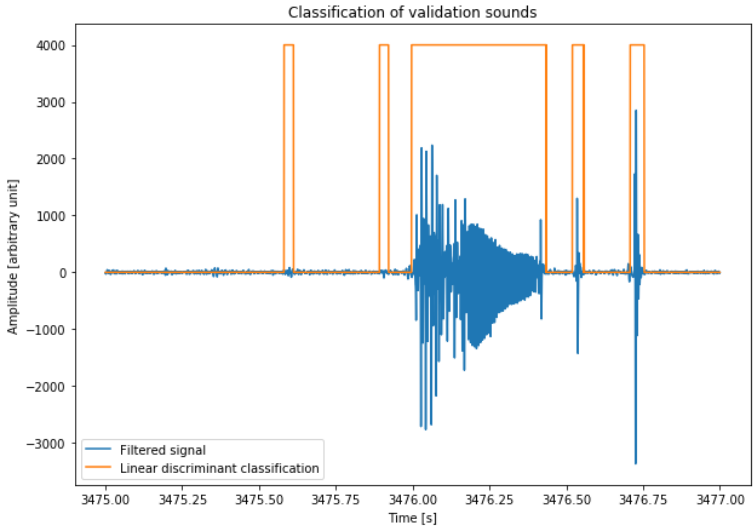

Figure 5.8: Prediction results for the validation set. Class label value has been multiplied by 4000 for visibility.

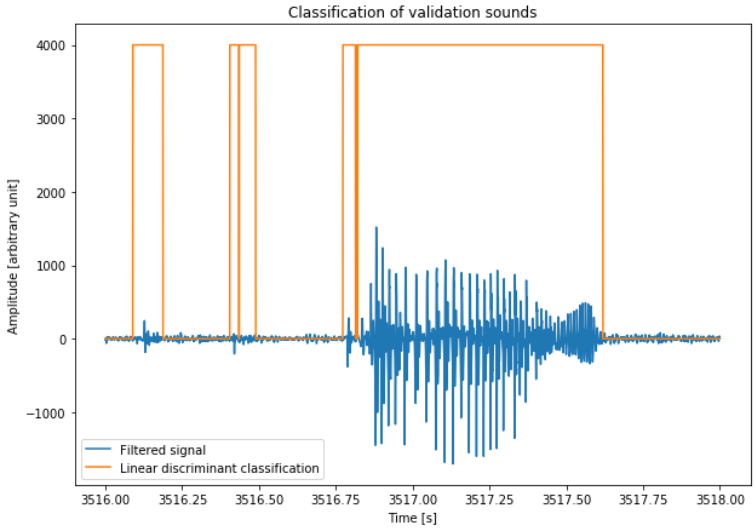

Figure 5.9: Prediction results for the validation set. Class label value has been multiplied by 4000 for visibility.

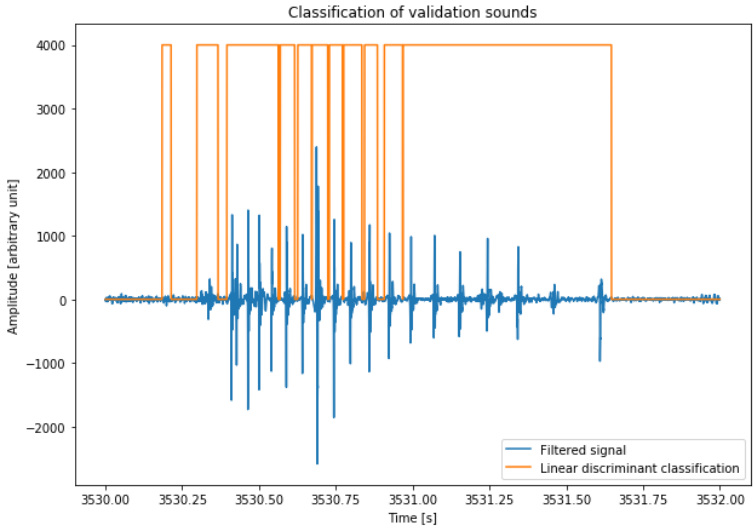

Figure 5.10: Prediction results for the validation set. Class label value has been multiplied by 4000 for visibility.

CHAPTER

6

CONCLUSION

A relevant discussion resulting from the differences in performance between the classifiers in section 5.2 and 5.1, is whether the former is qualitatively better than the latter. The quantified values in tables 5.2 and 5.4 suggests that the classifier performs better when a feature selection is performed before training. However, as shown in section 5.2.2, the resulting classification might become too sensitive, and possibly detect a large number of false bowel sounds.

The lack of medical expertise makes it difficult to determine whether the classifier in section 5.2 wrongly detects bowel sounds, or if these signal portions do in fact contain bowel sounds of very low volume. In addition, the problems concerning correctness of the training labels, which are discussed in section 5.1.2, do also very much apply to the classifier in section 5.2. A more consistent and correctly marked data set might yield fewer false positives for the classifier with feature selection.

In conclusion, a review of the results should be done by medical experts, with the final application in mind. Maybe is it more desirable to have a classifier which misses some of the existing bowel sounds, but provides a high degree of reliability for the bowel sounds it does detect (high precision). Or conversely, maybe it is more important not to overlook a present bowel sound, but a higher rate of false positives are acceptable. As mentioned in section 1.1.4, the motivation behind this bowel sound detection system is further integration into a meal detection system in an artificial pancreas. Section 2.1.3 referred to several sources stating that the number of bowel sounds recorded in a patient increases after intake of food, compared to when the patient is in a fasting state. This means that meal detection algorithm will only have to distinguish a relative increase of recorded bowel sounds. Thus,

the most important test is whether or not the detection algorithm contributes to the finished meal-detection system sensing a change in bowel sound occurrence rate. Testing whether the developed algorithm manages such a task was beyond the scope of this report.

The results presented in chapter 5 are certainly not perfect, but still very promising for the intended use. Metrics for both classifiers are in the higher end of the scale, something which is reflected in the classification of the validation sounds in sections 5.1.2 and 5.2.2.

The most serious problem is the ambiguity of what the algorithm should in fact detect. Fragmentation of sounds such as in figure 5.5 could yield unexpected results when counting the number of detected sounds. The splitting of the sound in figure 5.5 could be viewed as acceptable, since the spikes are quite clearly separated by low-amplitude signal, while fragmentation of the sound in figure 5.4 is more unexpected. Such seemingly random separations of coherent sounds might be very unfortunate for a counting procedure using the detection algorithm. In addition, the low-amplitude signals classified as ”bowel sound” in section 5.2 may not be preferable to detect. Such wrongful class predictions will also obviously sabotage for a correct bowel sound count. Suggestions to improvements of the algorithm, which might solve these problems, are presented in section 6.1.

All in all, however, the use of a linear SVM in combination with some subset of the features presented in table 3.1 produce promising results. This strategy is therefore a good foundation for further refinement of the bowel sound detection algorithm.

6.1 Further work

Several improvements and extensions could be done to the detection algorithm, which did not make it into this report. The first and most pressing matter is to establish clear guidelines for what needs to be detected by the classifier. Universal rules must be determined in consultation with medical experts, such that it is easy to manually decide whether a bowel sound has been encountered, and if the signal at hand consists of several consecutive sounds or one single sound. Said rules should then be reflected in the labels of the training set, such that the classifier is given consistent instructions. This might solve the issue of the classifier sometimes splitting up coherent sounds, as well as the questionable classifications of low-amplitude signals. If the sounds are still classified in a suboptimal way, post processing of the classification results could ensure that multiple sounds detected within a set time interval are combined into one single sound.

The data set provided to the algorithm could also be subject to certain improvements. Ideally, the data should be marked by medical experts with a certain experience in the field of auscultation, such that the training and test data is as accurate and consistent as possible. A larger data set could also improve classifier performance. To make the algorithm as robust as possible, the classifier should also be trained on data sets from several subjects. This would incorporate interpersonal differences in bowel sounds to the pattern recognition model, and hopefully yield a more versatile detection algorithm.

Lastly, if the presented algorithm is to be used in a finished artificial pancreas system, several considerations and modifications would be needed. The program would in this case run on an embedded platform, limiting both processing and memory resources. The detection would happen in real time, which introduces several requirements to code execution such as execution time restrictions and safety and fault tolerance considerations.

BIBLIOGRAPHY

[1] Healthline. 2018. How insulin and glucagon work. [Accessed: 2018-03-13]. URL: https://www.healthline.com/health/diabetes/insulin-and-glucagon.

[2] International Diabetes Federation. 2018. About diabetes. [Accessed: 2018-03-13]. URL: https://www.idf.org/about-diabetes/what-is-diabetes.html.

[3] Diabetes UK. 2018. Diabetes: the basics. [Accessed: 2018-03-13]. URL: https://www.diabetes.org.uk/diabetes-the-basics.

[4] American Diabetes Association. 2018. Living with diabetes: Insulin & other injectables. [Accessed: 2018-03-13]. URL: http://www.diabetes.org/living-with-diabetes/treatment-and-care/medication/insulin/.

[5] Health24. 2018. [Accessed: 2018-03-13]. URL: https://www.health24.com/Medical/Diabetes/Type-1-diabetes/Insulin-pump-may-cut-risk-of-heart-disease-deaths-for-diabetics-20150703.

[6] U.S. National Institute of Diabetes and Digestive and Kidney Diseases. 2018. Low blood glucose (hypoglycemia). Accessed: 2018-03-14. URL: https://www.niddk.nih.gov/health-information/diabetes/overview/preventing-problems/low-blood-glucose-hypoglycemia.

[7] Mamun, K. A. A. & McFarlane, N. 2016. Integrated real time bowel sound detector for artificial pancreas systems. *Sensing and Bio-Sensing Research*, 7, 84 – 89. URL: http://www.sciencedirect.com/

science/article/pii/S2214180416300058, doi:https://doi.org/10.1016/j.sbsr.2016.01.004.

[8] Gingras, V., Taleb, N., Roy-Fleming, A., Legault, L., & Rabasa-Lhoret, R. 2018. The challenges of achieving postprandial glucose control using closed-loop systems in patients with type 1 diabetes. *Diabetes, Obesity and Metabolism*, 20(2), 245–256. URL: http://dx.doi.org/10.1111/dom.13052, doi:10.1111/dom.13052.

[9] Artificial Pancreas Trondheim. 2018. Main page. [Accessed: 2018-05-30]. URL: http://www.apt-norway.com/.

[10] Healthline. 2017. Abdominal (bowel) sounds. Accessed: 2018-03-19. URL: https://www.healthline.com/health/abdominal-sounds#other-symptoms.

[11] Kölle, K. Untiteled. in progress.

[12] Cannon, W. B. 1905. Auscultation of the rhythmic sounds produced by the stomach and intestines. *American Journal of Physiology-Legacy Content*, 14(4), 339–353.

[13] Ching, S. S. & Tan, Y. K. 2012. Spectral analysis of bowel sounds in intestinal obstruction using an electronic stethoscope. *World Journal of Gastroenterology: WJG*, 18(33), 4585.

[14] Bray, D., Reilly, R., Haskin, L., & McCormack, B. 1997. Assessing motility through abdominal sound monitoring. In *Engineering in Medicine and Biology Society, 1997. Proceedings of the 19th Annual International Conference of the IEEE*, volume 6, 2398–2400. IEEE.

[15] Ranta, R., Louis-Dorr, V., Heinrich, C., Wolf, D., & Guillemin, F. 2010. Digestive activity evaluation by multichannel abdominal sounds analysis. *IEEE Transactions on Biomedical Engineering*, 57(6), 1507–1519.

[16] Tomomasa, T., Morikawa, A., Sandler, R. H., Mansy, H. A., Koneko, H., Masahiko, T., Hyman, P. E., & Itoh, Z. 1999. Gastrointestinal sounds and migrating motor complex in fasted humans. *The American journal of gastroenterology*, 94(2), 374.

[17] Craine, B. L., Silpa, M., & O'toole, C. J. 1999. Computerized auscultation applied to irritable bowel syndrome. *Digestive diseases and sciences*, 44(9), 1887–1892.

[18] Yoshino, H., Abe, Y., Yoshino, T., & Ohsato, K. 1990. Clinical application of spectral analysis of bowel sounds in intestinal obstruction. *Diseases of the Colon & Rectum*, 33(9), 753–757.

[19] Henry, N., Paul, N., & McFarlane, N. 2013. Using bowel sounds to create a forensically-aware insulin pump system. In *HealthTech*.

[20] Kreyszig, E. 2010. *Advanced Engineering Mathematics*. John Wiley & Sons. URL: https://books.google.no/books?id=UnN8DpXI74EC.

[21] Proakis, J. G. & Manolakis, D. G. 1996. *Digital signal processing: principles, algorithms, and applications*. Pearson Prentice Hall, 3 edition.

[22] What-When-How. 2018. Sampling. Accessed: 2018-05-03. URL: http://what-when-how.com/a-software-defined-gps-and-galileo-receiver/signals-and-systems-gps-and-galileo-receiver-part-3/.

[23] Chui, C. K. 1992. *An Introduction to Wavelets*. Academic Press Professional, Inc., San Diego, CA, USA.

[24] Huang, N. E., Wu, Z., Long, S. R., Arnold, K. C., Chen, X., & Blank, K. 2009. On instantaneous frequency. *Advances in Adaptive Data Analysis*, 01(02), 177–229. URL: https://www.worldscientific.com/doi/abs/10.1142/S1793536909000096, arXiv:https://www.worldscientific.com/doi/pdf/10.1142/S1793536909000096, doi:10.1142/S1793536909000096.

[25] Chegg Study. 2018. Ideal low-pass filter. Accessed: 2018-04-18. URL: https://www.chegg.com/homework-help/definitions/ideal-low-pass-filter-4.

[26] Butterworth, S. 1930. On the theory of filter amplifiers. *Wireless Engineer*, 7(6), 536–541.

[27] Wikipedia. 2016. By geek3 - own work: Butterworth filter. Accessed: 2018-04-18. URL: https://commons.wikimedia.org/w/index.php?curid=53970420.

[28] Zhao, Z., Pan, M., & Chen, Y. June 2004. Instantaneous frequency estimate for non-stationary signal. In *Fifth World Congress on Intelligent Control and Automation (IEEE Cat. No.04EX788)*, volume 4, 3641–3643 Vol.4. doi:10.1109/WCICA.2004.1343274.

[29] Starfish Medical. 2018. Hilbert signal processing graph. Accessed: 2018-05-29. URL: https://starfishmedical.com/assets/Hilbert-signal-processing-graph.png.

[30] Duda, R. O., Hart, P. E., & Stork, D. G. 2000. *Pattern Classification*. Wiley-Interscience, New York, NY, USA, 2 edition.

[31] Dyrdal, I. 2017. Mœnstergjenkjenning - forelesningsnotater til faget unik4590/unik9590/ttk4205. http://kybernetikk.unik.no/unik4590-ttk4205/Kopier/Unik4590Notater.pdf.

[32] Cortes, C. & Vapnik, V. Sep 1995. Support-vector networks. *Machine Learning*, 20(3), 273–297. URL: https://doi.org/10.1007/BF00994018, doi:10.1007/BF00994018.

[33] Scikit Learn. 2018. sklearn.metrics.confusion_matrix documentation. Accessed: 2018-05-31. URL: http://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html.

[34] Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. Jan 2002. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1), 389–422. URL: https://doi.org/10.1023/A:1012487302797, doi:10.1023/A:1012487302797.

[35] Scikit Learn. 2018. 1.13.3. recursive feature elimination. Accessed: 2018-06-02. URL: http://scikit-learn.org/stable/modules/feature_selection.html#rfe.

[36] Python Software Foundation. 2018. About python. Accessed: 2018-05-07. URL: https://www.python.org/about/.

[37] SciPy. 2018. About scipy. Accessed: 2018-05-08. URL: https://scipy.org/about.html.

[38] Scikit-learn. 2018. Scikit-learn home. Accessed: 2018-05-08. URL: http://scikit-learn.org/stable/index.html.

[39] SciPy. 2018. scipy.signal documentation. Accessed: 2018-05-08. URL: https://docs.scipy.org/doc/scipy-0.14.0/reference/signal.html.

[40] Bhattacharyya, S., Biswas, A., Mukherjee, J., Majumdar, A. K., Majumdar, B., Mukherjee, S., & Singh, A. K. Dec 2011. Feature selection for automatic burst detection in neonatal electroencephalogram. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 1(4), 469–479. doi:10.1109/JETCAS.2011.2180834.

[41] Scikit-learn. 2018. Api reference – scikit-learn. Accessed: 2018-05-26. URL: http://scikit-learn.org/stable/modules/classes.html.

[42] Scikit-learn. 2018. Scikit-learn algorithm cheat-sheet. Accessed: 2018-05-26. URL: http://scikit-learn.org/stable/tutorial/machine_learning_map/.