CHALMERS
UNIVERSITY OF TECHNOLOGY

Machine learning for symptoms quantification of Parkinson's disease patients

Master's thesis in Biomedical Engineering

CECILIA HERNQVIST & MATILDA ROSANDER

Department of Electrical Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2017

MASTER'S THESIS EX050/2017

Machine learning for symptoms quantification of Parkinson's disease patients

CECILIA HERNQVIST & MATILDA ROSANDER

Department of Electrical Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2017

Machine learning for symptoms quantification of Parkinson's disease patients
CECILIA HERNQVIST & MATILDA ROSANDER

Supervisors: Anders Ericsson & Amin Ojani, Acreo Swedish ICT AB, Gothenburg, Sweden
Examiner: Tomas McKelvey, Department of Electrical Engineering, Chalmers University of Technology, Gothenburg, Sweden

Master's Thesis EX050/2017
Department of Electrical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Phone number: +46 31 772 1000

Gothenburg, Sweden 2017

Machine learning for symptoms quantification of Parkinson's disease patients
CECILIA HERNQVIST & MATILDA ROSANDER
Department of Electrical Engineering
Chalmers University of technology

Abstract

Parkinson's disease is a disease affecting the nervous system, with about 20 000 people diagnosed in Sweden in total. The most significant symptoms influence the motor skills, which may greatly lower the quality of the patients' life. The effect of the disease is mitigated by medication, but adjusting the medication dose is difficult and therefore a tool for simplifying the process is needed. Besides this, diagnostics regarding the severity of symptoms often differ between individual physicians. The purpose of this thesis is to help develop a diagnostics tool by using machine learning to classify the severeness of a symptom associated with the disease called bradykinesia. The data used in the project was gathered from gyroscope and accelerometer sensors, which were attached to both the wrists and ankles while the patient conducted two prescribed movement activities; continuous heel tapping and hand rotation. Twenty different machine learning models were trained and tested for the classification based on the UPDRS scale (Unified Parkinson's Disease Rating Scale). The two most promising models were the support vector machine together with a cubic and a polynomial kernel, respectively, where the cubic kernel is a special case of a polynomial kernel. To train the models, at set of 164 features were extracted from the data, whereas different hypotheses were formed based on subsets of the original feature set. It was found that the use of all available features from both the hand and foot data generally gave the best results. One promising outcome was regarding classification between healthy subjects and patient signals deemed as symptom-free, where several models were able to distinguish between the two classes. This result indicates that machine learning algorithms are able to detect signal characteristics not noticeable to the visual inspection of the physician.

Keywords: Signal processing, statistical learning, Parkinson's disease, symptom quantification, UDPRS, machine learning, bradykinesia

Acknowledgement

The authors of this thesis would like to thank Acreo for the opportunity to conduct this project regarding symptom quantification of Parkinson's disease patients. It has been our pleasure to be introduced to the working environment and all the knowledge of Acreo. Especially, we are grateful towards our supervisors, Anders and Amin, for the continuous feedback and support during our work. The outcome of this thesis has been made possible by their involvement and encouragement. A deep thank you should also be directed towards our examiner Tomas, for all feedback during the process, which forced us to reflect deeper and consider other aspects.

Cecilia Hernqvist & Matilda Rosander, Gothenburg, June 1st, 2017

Contents

List of Figures

List of Tables

Abbreviations

ADL	Activities of Daily Living
DBS	Deep Brain Stimulation
kNN	k-Nearest Neighbor
MAD	Median Absolute Deviation
MARS	Multivariate Adaptive Regression Splines
MSE	Mean Square Error
Musyq	Multimodal Motor Symptoms Quantification platform for individualized Parkinson's disease treatment
PCA	Principal Component Analysis
RMS	Root Mean Square
RSS	Sum of Squared Residuals
SE	Standard Error
SVM	Support Vector Machine
UPDRS	Unified Parkinson's Disease Rating Scale

1 Introduction

Today there are about 20 000 persons suffering from Parkinson's disease in Sweden, most of them are 50 years or older. The characteristics of these patients is usually that their movement pattern is distinctive and oppressive, if not treated right with medication. For the physicians to be able to help these patients, a correctly adjusted medication is necessary. This is hard to achieve since there are no way of measuring the severeness of the motor symptoms other than listening to and observing the patient. This especially becomes a problem for long-term patients since the effect of the medication is wearing off after several years of usage. If the dose is too low, the symptoms will still be present, and if the dose is too high the patient may become hypermobile. A tool that would simplify the process of adjusting the medication and making the dose well suited for the particular patient is needed. This also induce the need for objective observation outside of the short meeting between the patient and the physician. The scope of this thesis work is a part of a larger project trying to develop easier and more objective tools for Parkinson's disease diagnosis.

1.1 Background

Acreo is part of a project called Multimodal motor symptoms quantification platform for individualized Parkinson's disease treatment (Musyq), where the main purpose is to increase the quality of Parkinson's diagnostics by means of more precise sensors and ways of quantifying movement data [1]. The work described in this master's thesis is a part of this project. Today, symptoms are evaluated and quantified by physicians with help of patient journals and through observation. However, most patients struggle to report symptom severity and nature of symptoms accurately, especially since the patient will often try to report symptoms from earlier days, and therefore their memory of symptoms might have been altered from how it really was. Acreo has collected accelerometer and gyroscope measurements of repeated turning of the wrists one after the other, and heel tapping respectively of 41 individuals; 21 healthy and 20 with Parkinson's disease. Difficulties with these specific movements are characteristic for Parkinson's disease, implying that its analysis may yield an insight into possibilities for quantitative evaluation of symptoms.

1.2 Purpose

The main focus of this project involves using machine learning to quantify the raw data collected from the ankle and wrists, and train different classification algorithms to evaluate their performance with respect to diagnosing Parkinson's based on the Unified Parkinson's Disease Rating Scale (UPDRS) [2]. Trials will also be done to see if the classification algorithms are able to classify between healthy subjects and Parkinson's disease signals that has been classified with UPDRS 0 from the physicians, meaning they are visually symptom free. This is done to see if there are underlying characteristics in the data that might indicate that a signal belongs to a Parkinson's disease patient even though the particular signal is deemed to be without symptoms. The results from this thesis will be used by Acreo to find suitable measures to further classify Parkinson's disease, especially when it comes to continous observation during everyday movements which could be achieved by sensor bracelets or similar add-ons. This kind of more objective and more frequent assessment of symptoms would be of great help for rating a patient's condition as well as for the evaluation of changes of a patients treatment. Today when a physician is evaluating a patient with

possible Parkinson's disease, the physician have to base his or her judgment on the very short time frame during the meeting, as well as the patients' own perceived symptoms which might be biased. Additionally, similar symptoms may be rated differently depending on the physicians training and experience as well as personal factors. The aim of this thesis is to quantize the evaluation of Parkinson to avoid differences in rating and assessment.

1.3 Limitations

Parkinson is a disease with multiple possible symptoms, both physical and psychological symptoms. This project will only take physical motor symptoms into consideration, with the main focus on bradykinesia.

1.4 Research questions

The following research questions was formulated, to distinguish the main concerns to be investigated during this project.

1. Does a certain feature subset work better for the prediction of UPDRS?
2. Are different feature subsets suitable for feet sensor data compared to wrist sensor data?
3. Does data collected from the feet or wrists work better for prediction of UPDRS? Or foot and wrist data together?
4. Which method(s) work best for the prediction?

2 Theoretical reference frame

In the following chapter the theoretical background for this project is given. First a description of Parkinson's disease and its existing treatments are presented, then some earlier research are discussed and finally a short introduction to machine learning is given.

2.1 Parkinson's disease

Parkinson's disease emerges due to the fact that nerve cells in substantia nigra, a part of the brain which is important for the reward system of the brain, are degenerating. The cells responsible for producing the hormone dopamine seems to be exposed to the breakdown to a higher extent. Beside having an impact on the human feelings, the dopamine also controls the movements of the muscles [3]. In most of the cases the reason for Parkinson's disease is age, the older a person is the higher the risk of getting the disease, in all humans the production of dopamine decreases with age. Other reasons can be heredity, environment, or some environmental toxins that increases the risk of getting Parkinson's disease, for instance hydrocarbon solvents. An injury to the head may also play a role, and some injuries like stroke has symptoms that resembles Parkinson's disease [4].

The symptoms of Parkinson's disease usually appear slowly and gets worse over time. The first symptom to emerge that is detectable is usually tremor. It can start in the finger of one hand and later spread to the whole hand and so on. Other symptoms can be bradykinesia and akinesia, bradykinesia means that the motions are slow, sometimes the muscles might even freeze, and akinesia is when the muscles get stiff and the patient has a hard time initiating the movement [3]. A symptom that differentiate a bit from the others is depression, a Parkinson patient is very likely to be affected by depression. The cause for that is probably a combination of them having a chronic disease and that the neurochemistry of the brain has changed. Research proposes that the depression can be an effect directly from the disease's chemical influence on the brain, the affected area of the brain does not only control the movements of the body but also the mood. As mentioned before the production of dopamine is decreasing, but other hormones are also decreasing such as serotonin and norepinephrine which also are involved in the mood, motivation and apetite among other. The frontal lobe is underactive in a patient with Parkinson's disease, which is a part of the brain that is important in controlling the mood [5].

Today, there is no way of curing Parkinson's disease, but there is medication, the most common one is called Levodopa. Levodopa is an amino acid that exists naturally in the body and it is an initial stage to dopamine. Dopamine cannot be used as medication since it cannot pass the blood-brain barrier. The dose is low in the beginning of the treatment and then gradually increasing until the ideal dose for the particular patient is reached. The medication will give effect after some weeks, but in some cases it can take several months until any difference can be noticed. Since the time that Levodopa is relieving the symptoms of the disease is relatively short it is usually combined with a dopamine amplifier which blocks the enzymes that breaks down dopamine, this makes the effect of the Levodopa last longer [6].

The Levodopa treatment can help the patient live with the disease for many years with only some inconvenience. After some years problems usually emerges, the effect of the medication becomes irregular. There will be periods of dyskinesia and periods where the medication does not give any or little effect. This can be prevented partly by regulating the doses of Levodopa, but when this stops working the patient is in what is referred to as

an advanced phase of the disease. When this stage is reached it is time to start thinking about alternative ways of treating the patient. One way could be to use a pump with a dopamine agonist, apomorphine. This is a treatment that was developed by neurologists in England in 1980. Apomorphine is injected under the skin in the subcutaneous tissue where it is absorbed and transported to the brain, the effect of apomorphine is the same as dopamine's. Apomorphine is preferred since a more even effect will be achieved if it is steadily injected [6].

Another alternative is to have a pump with Duodopa, a Levodopa/carbidopa-gel, it was developed in Sweden and is used in Europe since 2004. The gel is pumped directly into the small intestine where it is easily absorbed by the blood. As for the pump with dopamine agonist, this pump makes the flow of the medication even, and therefore also the effect of the medication will be more even. The third alternative of treatment when the disease has entered the advance stage is Deep Brain Stimulation (DBS), a method that has been used since 1993. It is based on the discovery that high-frequency current can affect the brains different functions, this can be done with electrodes. The effect of DBS recalls the effect of Levodopa, mainly the motor symptoms is reduced, and the difference from Levodopa is yet again that the treatment is more continuous. DBS also has a better effect on tremor [6].

2.2 The Unified Parkinson's Disease Rating Scale

The Unified Parkinson's Disease Rating Scale (UPDRS) is used to evaluate the patients disease and was developed in the 1980's. The test consists of six different parts, part 1: Mentation, Behaviour and Mood, part 2: Activities of Daily Living (ADL), part 3: Motor Examination, part 4: Complications of Therapy, part 5: Modifies Hoehn and Yahr Staging and part 6: Schwab and England Activities of Daily Living Scale [7]. The first three parts and half of the fourth are graded with on a scale from 0 to 4, where 0 corresponds to normal and 4 corresponds to severe symptoms. The other parts are answered with Yes or No questions [7]. The Motor Examination activities are especially developed to classify the symptom of bradykinesia. The heel tapping test used for the study behind this work belongs to that part and corresponds to item 26 in the UPDRS rating protocol. This test examines the agility of the leg by instructing the subject to tap their heel against the floor repeatedly, with an amplitude of at least 7.5 cm. The hand turning test also belongs to part 3: Motor Examination and corresponds to item 25. The task given to the patient is to rotate his or her hand rapidly, changing between a pronation and a supination position of the hand [7]. The physician then rates the test on the scale from 0 to 4 based on how well the subject is able to complete the task. The five-step rating increases depending on the severity of the patients ability to perform the heel tapping and contains the following rating steps: 0 = normal, 1 = Mild slowing and/or reduction in amplitude, 2 = Moderately impaired. Definite and early fatiguing, 3 = Severely impaired. Frequent hesitation in initiating movements or arrests in ongoing movement and 4 = Can barely perform the task [7].

When the data for this study was collected, the subjects were filmed. This was later displayed to three clinicians independently; All three individually examined the hand data and two of them also examined the feet data. They evaluated the patient by using the UPDRS rating. The resulting UPDRS score from these evaluations are used as the ground truth during the classification process done by the machine learning models.

2.3 Earlier research

In the earlier studies working with quantification of Parkinson's disease the researchers have taken different approaches in selecting relevant data for feature generation, but a lot is also similar. The main resemblance is extracting features by using knowledge about movements and motion patterns in a person affected by Parkinson's disease. Bonato et. al. stated that high frequency components could be analysed from the signals, since these are often generated by rapid movements such as fast acceleration or deceleration. They chose to work with both linear and non-linear features, such as energy of the signal, amplitude, dominant frequency (linear), as well as entropy values and correlation dimension estimates (non-linear) [8]. Entropy values as a feature was also used by Cancela et. al., together with a measurement of the root mean square (RMS) of the frequency power spectrum and the range of the values [9]. Parts of these three features were combined in different ways to form the training set, such as combining the RMS with the sample entropy and the range [8].

Similar features were chosen by Patel et. al., who in their analysis of bradykinesia and dyskinesia symptoms also looked at the root mean square, as well as value and range of each signal in each sensor direction. In the frequency domain they also investigated features such as dominant frequency, energy associated with the dominant frequency, and the entropy [10]. Tsipouras et. al. used similar features as the ones mentioned, but they also looked at the standard deviation of the signal as well as the mean value [11]. Common for a lot of the studies mentioned is that they use windowing to separate the signals into smaller frames, where features are extracted both from the smaller frames as well as from the signal as a whole.

2.4 Machine learning

Machine learning is the name for a large variety of techniques used to allow a computer to learn and predict responses based on a set of data. This means that the computer should be able to predict answers without being explicitly programmed what answer to give, rather it should be trained to make a qualified decision based on earlier experiences with the training data. There are three main problems which machine learning are being applied to; namely regression, classification and clustering. Regression is the problem of predicting a continuous response (such as a value between 0 and 5), whilst classification is the problem of predicting a qualitative categorical response (such as classifying something as blue, green, or red). Clustering is the problem of grouping variables to find patterns in the data and assigning the data to different groups [12].

When using machine learning, the commonly used strategy is to use one set of observations to train the model and tune its parameters (training set), and another set of observations to validate the functionality and accuracy of the model (test set). There are numerous different methods to apply in machine learning, and from a lot of the traditional methods additional alternative methods have emerged as improvements of the old ones, or more adapted to a certain kind of problem. Different methods may be grouped depending on the different approaches with which the learning is obtained. Some of these groups are nearest neighbour methods, decision tree induction, error back propagation, reinforcement learning and rule learning [13].

One pitfall to avoid in machine learning is the problem of overfitting, which might occur when the model becomes too specific with respect to the training data. In that case a good result will not be achieved when validating the model on a test set, since the model has been trained too much on certain characteristics available in the training data but not in

the test data. Therefore a trade off has to be considered, between a large enough feature set to learn the overall characteristics of the data, but still not too large to start distinguishing between different objects of the same class [14]. Another way to put this is that it is desired to make the model general, but not too general since that would mean that the model shows poor performance when trying to classify, which means that the model is underfitted. The problem of overfitting and underfitting goes hand in hand with the bias-variance tradeoff. High variance is when a model depends greatly on the training data, e.g. is overfitted. Therefore the appearance of the model might change drastically when introducing different data. High bias is when the model has a large fitting error to the training data, i.e. the model is too rigid to give a good estimate of the data [15], e.g. underfitted. The optimum goal is to keep both the bias and the variance low, but this might be extremely hard, or impossible to achieve fully. This is because for most cases the variance is increased with decreased bias, and vice versa [14]. To counter this, one needs to choose a complex enough model to avoid too much bias, but avoid making the model too complex to avoid overfitting and variance. The number of features for each observation might also influence the performance of different models. One expression often used in machine learning is the "Curse of dimensionality", which should be considered especially for non-parametric methods. One example is k-nearest neighbors which aims to classify based on a set of k neighbors in the feature space. With a high number of features, the nearest neighbors might be far away and this causes the curse of dimensionality [16]. Therefore it is important to investigate important features that improve the model as well as using valid methods for feature selection to avoid using a too high-dimensional feature set.

3 Method

Within this section, the process of the project is described. The procedures taken are presented as well as why certain decisions have been made.

3.1 Available data

The data that has been collected in the Musyq project are recorded from 41 subjects, whereas 21 are healthy persons and 20 are patients diagnosed with Parkinson's disease. The healthy controls were all in the age range between 50 to 76 years old, while the age range of the Parkinson's disease patients were 61 to 82 years old. All subjects gave their informed consent to the study [17], and data was recorded for the limbs while conducting several movements from the UPDRS rating protocol concerning motor ability. One of these evaluation items is to do a heel tapping, in accordance to item 26 of the UPDRS rating protocol [7]. The data was recorded for both ankles and both wrists, but only one limb was moved at a time. All healthy subjects did 8 repetitions of each test sequence, whilst Parkinson's disease patients were allowed to do more repetitions if possible. For these subjects mostly repetitions between 10 - 14 times were gathered. The subjects diagnosed with Parkinson's disease had not taken any medication before the first trial, and it was therefore expected to see more signs of bradykinesia during the first trials for each patients, if this was a symptom they had experienced. Before the second trial they received a dose 1.5 times higher than their original dose, which for some patients resulted in dyskinesia present after the intake. After a few hours the effect from the medication was expected to have worn off, which might result in bradykinesia being more present again during the later trials. Time zero was considered at the first dose intake, e.g. before the second trial. At the same time as the dose intake, a blood sample was also collected. More blood samples were collected at later times as well to measure Levodopa and other drugs present in the blood [17].

For this study, Acreo used sensors called Shimmer3 [18], which are commercially available and contains both a 3-axial gyroscope and a 3-axial accelerometer. During the test each subject had a sensor placed on both ankles and both wrists collecting the movement in the X-, Y-, and Z-direction.

The data were collected at a sampling frequency of 102.4 Hz and depending on how many repetitions that were done, there is around 3 - 6 hours of data gathered for each subject. The raw data was examined by Acreo after the study was finished to split it, based on visual inspection, into the different trials for easier processing. The data files available for this work contain all the different trials for each subject with the measurements in all three directions for both gyroscope and accelerometer, as well as a fused measurement calculated by taking the magnitude for all directions for each time sample:

$$|x| = \sqrt{f_x^2 + f_y^2 + f_z^2} \quad . \tag{1}$$

where $|x|$ is the magnitude for each time sample and f_x, f_y, and f_z are a time sample for each signal in each of the three directions. The different UPDRS scores given by the physicians are also available for each of the limbs. In the case that a patient was deemed to have more severe symptoms in one side of the body, the physicians gave more weight to the rating of the more affected side. How well a subject is able to perform the assigned heel tapping is reflected on the signal, Figure 1 and 2 displays one trial for a healthy control subject and a subject affected by Parkinson's disease, respectively. The displayed signal is

the magnitude of the accelerometer data, in accordance to Equation 1. Both the amplitude and the characteristics of the signal varies heavily between the two different subjects.

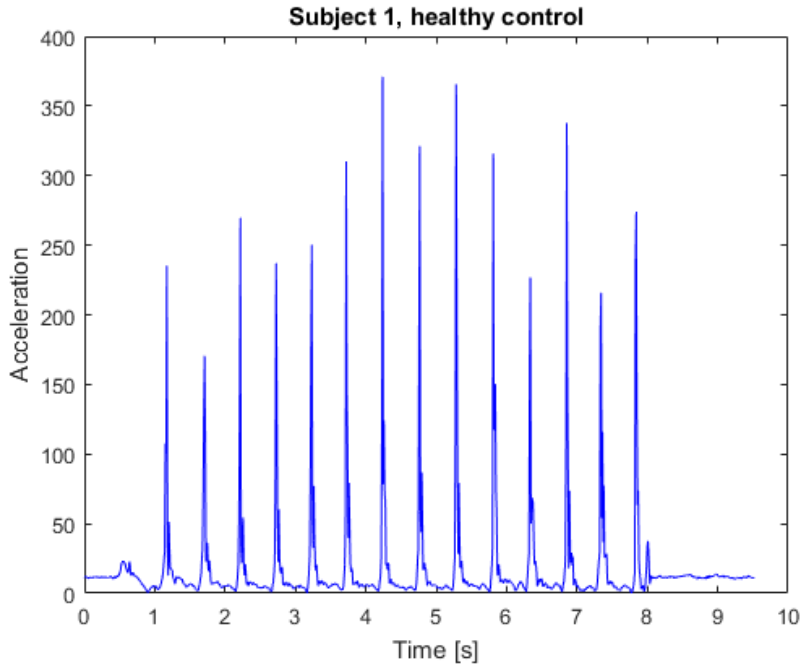


Figure 1: One trial of the heel tapping exercise from a healthy control subject, signal obtained from the absolute value of all three directions of the accelerometer data.

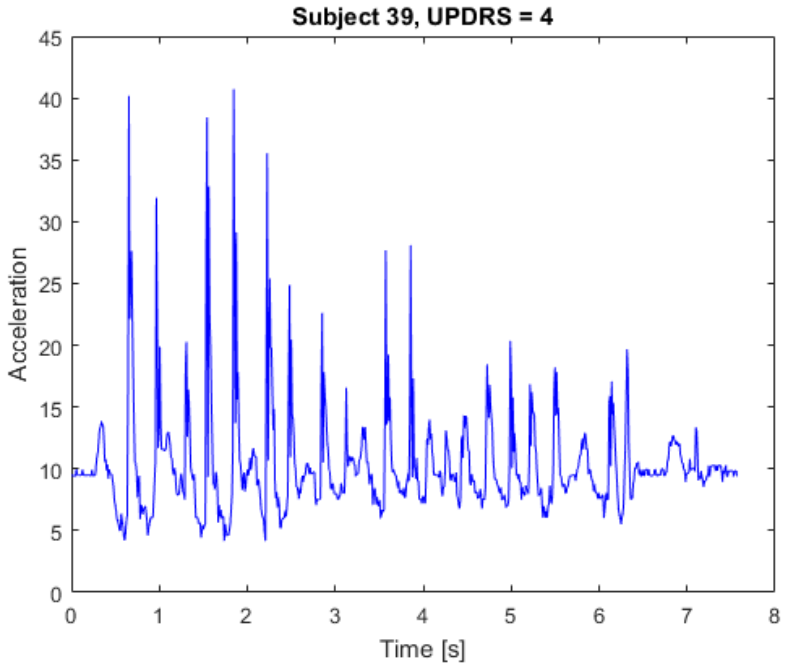


Figure 2: One trial of the heel tapping exercise from a subject affected by Parkinson's disease, with a UPDRS rating equal to 4. Signal obtained from the absolute value of all three directions of the accelerometer data.

3.2 Pre-processing of raw data

Before features can be extracted from the available data, some preprocessing was performed to prepare the data for further processing. Measures that were taken was removal of faulty signal data and extraction of windows for the signal. The UPDRS scores from the different physicians was also merged to one value, by taking the mean of the UPDRS score from the individual physicians.

3.2.1 Removal of faulty data

Due to some inconsistent measurements during the study, some data had to be discarded. For a few trials, the signal simply has not been recorded due to some sensor failure. This was the case for subject 20 where all data had to be discarded due to the sensors not working during the trials. For subject 6 the first trial on both feet were deemed erroneous since the amplitude for the first trial was suspiciously lower than the other trials, even though subject 6 is a healthy control which should not suffer from too big changes in amplitude from trial to trial. For subjects 7, 9, 10, 14, 15, 22, 25, 26, 30 and 34 signals from one or a couple of trials were missing for one foot due to problems with the sensors. For subject 35 all recordings of the left foot were missing. The last data that was discarded was all data from both feet of subject 29, which was a Parkinson's disease patient. However, during trials both clinicians deemed the patient as symptom free in both feet, which resulted in a UPDRS rating of 0 for all trials of the heel tapping. The data from this subject was therefore removed since it was a diagnosed patient, but without symptoms affecting the feet. For the hand data subjects 2, 4, 14, 22, 25, 26, 30, 31 and 34 had one or more signals missing. Subject 11 also had 6 signals recorded from the left hand that were discarded due to abnormal characteristics of the signals. Subject 40 also had extremely diverging data for all signals from the right hand. For subject 11 all the data from the left hand was missing. All data that were deemed as erroneous was discarded before further pre-processing could be accomplished. After this, measurements from 19 Parkinson's disease patients, and 21 healthy subjects were left.

3.2.2 Mean of UPDRS

Since a set of independent clinicians diagnosed the subjects participating in the study, the UPDRS scores given was analyzed to gain insight in how often the physicians agreed on a classification of the symptoms displayed by the patient. For both feet, the physicians gave the same score for 59 percent of the trials, whilst the other judgments varied. For the hands agreement were reached for 69.2 percent of the trials. The greatest variation was for one signal from subject 39, where the first clinician gave the subject the highest UPDRS score of 4, while the other clinician deemed it as the lowest score, 0. These differences in rating gives an indication of the importance of an objective and unbiased symptom classification.

As stated earlier, when diagnosing Parkinson's disease the physician gives greater weight to the more affected side of the body to decide on a suiting rating of the UPDRS scale. To get a fair comparison to real life diagnostics, only the most affected leg or hand of each patient will be used for further training of the machine learning methods in this work. To decide on this, a mean value for the UPDRS rating of each trial for each patient had to be calculated. The mean values for both legs and hands were then compared to only keep the most affected leg or hand, respectively. The mean value was also calculated to only have one UPDRS rating to compare to further on when training the models. Table 1 displays the UPDRS rating for each patient with consideration to the most affected leg and hand.

ID	Trials	UPDRS 0		UPDRS 1		UPDRS 2		UPDRS 3		UPDRS 4	
		Hand	Foot	Hand	Foot	Hand	Foot	Hand	Foot	Hand	Foot
7	25	1	3	8	9	4	0	0	0	0	0
8	18	0	0	6	14	8	0	0	0	0	0
11	15	0	5	2	6	2	0	0	0	0	0
12	24	0	5	7	5	4	1	1	1	0	0
13	24	0	6	5	5	7	1	0	0	0	0
14	26	0	1	3	11	10	1	0	0	0	0
15	11	-	6	-	5	-	0	-	0	-	0
16	26	0	12	2	1	11	0	0	0	0	0
29	14	0	-	6	-	8	-	0	-	0	-
30	28	0	1	3	13	11	0	0	0	0	0
31	28	0	0	0	0	1	6	13	8	0	0
33	30	0	1	0	13	10	1	5	0	0	0
34	24	0	1	0	11	8	0	4	0	0	0
35	20	0	0	0	2	0	4	5	4	5	0
37	28	0	0	0	8	14	6	0	0	0	0
38	28	0	8	6	6	6	0	2	0	0	0
39	20	0	0	6	0	6	0	2	5	0	6
40	28	0	11	7	3	7	0	0	0	0	0
41	28	0	1	0	5	2	6	12	2	0	0
Sum:	**445**	**1**	**61**	**61**	**117**	**119**	**26**	**44**	**20**	**5**	**6**

Table 1: The mean UPDRS rating for the most affected leg/hand of each patient. Displays the number of trials for each patient and how many of those trials that were deemed as UPDRS 0, 1, 2, 3 and 4, respectively.

As can be seen in Table 1, there are not a lot of representation in the higher scores on the UPDRS rating; there are only six signals that were scored with a UPDRS of 4 for the foot data, and they all belong to the same patient. Due to the lack of signals diagnosed with UPDRS 4, these signals are removed from this work, to avoid risking classification based on personal traits. After this a total of 224 trials (feet data) and 221 trials (hand data) from subjects affected by Parkinson's disease were stored and used in further work.

3.2.3 Window functions

One common procedure used in signal processing and when analyzing signals is to split the signal into several windows to be able to analyze one part of a signal at a time. There are several different windowing techniques used for different purposes.

When windowing a signal, it might be necessary, depending on the application, to force the signal to zero in the beginning and at the end of the time series to reduce the consequence of leakage. If this is not applied, it will result in a discontinuity of the signal, which will cause problems with the process of the Discrete Fourier Transform (FFT). This discontinuity might cause leakage to appear, which means that energy leaks from the signal to the side lobes. There are ways of avoiding leakage in theory, but they will not work in practice [19].

The solution to this is to use a so called smoothing window. The smoothing windows can have many different shapes, the difference between them depends on how they convert from the low weights near the edges to the higher weights near the middle of the sequence. Besides forcing the signal to zero at the end and beginning it also adds distortion to the

time series. This gives side bands in the spectrum which reduce the frequency resolution of the analyzer. By this, the smoothing window can suppress side lobes near the relevant frequency in the frequency domain [19].

For this particular project a Hamming window was used as a smoothing window, for all feature generation in the frequency domain. It is defined by the following equation:

$$w(n) = \alpha - \beta \cos \frac{n\pi}{M}, \quad for \quad -M \leq n \leq M, \tag{2}$$

where M is the length of the window and w is the window value. There are two coefficients, α and β, that were proposed by Richard W. Hamming as $\alpha = 0.54$ and $\beta = 0.46$. The shape of the Hamming window is similar to the shape of a cosine wave [19].

Besides using a Hamming window in preparation of generating features from the frequency domain, a rectangular window was also used. This is because using a smoothing window might not always be necessary when working in the time domain. A rectangular window in this aspect does not apply any weight to the signal, rather it cuts out a part of the signal as it is. The rectangular window is defined by the following expression [19]

$$w_R(n) = 1, \qquad -M \leq n \leq M \tag{3}$$

where M is the window length.

MATLAB has a ready-made function, simply called `hamming`, for implementation of a symmetric Hamming window, which uses Equation 2. For this function, the window size is handed as input to create the window. The window size for both windows were decided as 1 second, which in this case corresponds to about 75 samples. It was applied to all of the signals acquired from the accelerometer and the gyroscope, both for all of the singular directions (x, y, z) as well as for the combined signal based on the absolute value for all three directions, for the accelerometer and the gyroscope respectively.

The rectangular window was implemented in MATLAB by applying the built-in function `rectwin`. The Hamming window was used with a 50 percent overlap to reduce any loss of information. If not, important information may be lost due to the weighting function applied with a Hamming window, where samples at the ends of the window receives less weight than the central samples. When deciding on how big the overlap the should be, one always has to consider the computational expense that follows with a larger overlap. 50 percent is a commonly used overlap for Hamming windows [20]. Features were thereafter generated based both on the whole signals, windows extracted with the Hamming window, as well as windows extracted with a rectangular window, depending on what kind of feature that should be generated. Besides the six different signals for each trial (x-, y-, z-direction for accelerometer signals, x-, y-, z-direction for gyroscope signals), two merged signals was also produced by taking the vector magnitude of all three directions for the accelerometer and the gyroscope, respectively.

3.3 Feature generation

When generating features, it is important to distinguish different characteristics in the data that may contribute in recognizing a certain kind of signal. Some characteristics can be detected visually just by looking at a signal generated by a healthy person and one generated by a person diagnosed with Parkinson's disease. The main characteristic which is protruding is that a Parkinson's patient has a lower angular velocity while performing the task of heel tapping. In accordance to the symptom of bradykinesia, the initiation of the

heel tapping is also prolonged resulting in a longer endurance of the heel tapping, while a healthy person has a quick initiation and faster execution. Another main difference is that healthy subjects display a lower amplitude between each heel tapping, whilst Parkinson's patients have more of a flicker or disturbance between each tap. These observations are in accordance with a conclusion made by Griffiths et. al., where they mention that Bradykinesia is recognized by "fewer movements made with lower acceleration and amplitude and with longer intervals between movements" [21]. However, a visual inspection is not always enough in distinguishing patterns in the data, and other aspects both in the time domain and the frequency domain might have to be considered.

When choosing what features to generate for this thesis partly signal characteristics that can be detected visually was chosen. The main part however, was features generated in accordance to the preceding master thesis completed by Wendebourg [22]. The first four features described below might be considered more subjective since these are based on visual inspection of the data. The other described features are more objective since they are not derived based on the available data.

To extract features from the frequency domain some frequency analysis has to be done. In MATLAB Fourier transform was applied by using the built-in MATLAB function `fft`, which uses a fast Fourier transform algorithm to compute the discrete Fourier transform, which is defined by

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-\frac{2\pi k n}{N} j}, \tag{4}$$

where N is the total number of samples [23]. By analyzing the frequency spectra of the signals, one can see that the frequency content is distributed approximately between 0.75 - 4.5 Hz. This is a reasonable range since the common maximum human movement is seated around 3.3 Hz ([24],[25]). Both frequencies below and above 3.3 Hz is however frequently represented in patients with Parkinson's disease, since Bradykinesia and rigidity causes patients to move slower and with less strength, therefore their motor fluctuations are expected to be lower [25]. For the symptom of tremor, the frequency range is often between 4-6 Hz [26].

Besides only extracting the features, which are described in the subsequent sections, it might also be interesting to analyze how the different characteristics changes over time. Keijsers et.al. used the standard deviation of selected features when trying to classify dyskinesia while using neural networks as the classification algorithm [27]. Tsipouras et. al. also used standard deviation for feature generation to classify dyskinesia ([28], [11]). Therefore, the standard deviation of each feature was also applied by calculating each feature for several windows of each signal, and then calculating the standard deviation which measures the distribution of a given signal around its mean, or in this case, the distribution of features calculated for N windows around the mean value of all the features generated from the given signals. The general definition of standard deviation for discrete samples is given by the following equation [29]:

$$\sigma = \sqrt{\frac{1}{N} \sum_{n=0}^{N-1} \left(X[n] - \mu\right)^2} \tag{5}$$

where μ is the mean value and N is the total number of samples for the given signal. By substituting μ with its original summation and changing Equation 5 to be based on a feature

as done in this work, the standard deviation for one feature x is

$$\sigma = \sqrt{\frac{1}{M}\sum_{m=1}^{M}\left(x_m - \frac{1}{M}\sum_{m=1}^{M} x_m\right)^2} \tag{6}$$

where M denotes the total number of windows, m denotes the mth window and x_m is the feature calculated for the mth window. Since the features may vary a lot in size and kinds of units, one final step to ensure a valid comparison between the standard deviations for the different features, is to calculate the relative standard deviation. In order to calculate the relative standard deviation of the features in the frequency domain, the MATLAB function `spectrogram` was used, which returns the short-time Fourier transform (STFT) denoted as X_{STFT}, the frequency range f_{STFT} as well as the energy spectrum estimates E_{STFT} [30]. Goodwin defines the STFT for a sliding window function as the following equation [31].

$$X[s,k] = \sum_{n=0}^{N-1} w[n]x[n+s]e^{-i\omega_k n} \tag{7}$$

In this formula, w is the window and N is the window length. x is the given input signal, s is a discrete time variable and k is the frequency range. This notation could be rewritten as

$$X_{STFT}[s, f_{STFT}] = \sum_{n=0}^{N-1} w[n]x[n+s]e^{-i2\pi n f_{STFT}}. \tag{8}$$

The window function used in MATLAB to calculate the STFT was the Hamming window described in Section 3.2.3, which was implemented directly in the `spectrogram` function.

3.3.1 Number of subsequent movements

To see if there were some correlation regarding the endurance of performing the heel tapping or the hand turning, the number of subsequent movements defined as peaks were counted for each trial and used as one of the features. This was examined by using the vector magnitude of all three directions of the accelerometer sensor (as specified in Equation 1), where the movements are easily distinguished as high peaks of acceleration when visually inspecting the signal. When extracting this feature, the MATLAB function `findpeaks` were used. To make sure that the function only distinguished actual maximum movements, such as the heel tapping for feet data, a minimum peak height and a minimum peak distance had to be determined, as well as a minimum peak prominence to not judge a sub peak as an independent peak. The threshold for when a local maxima should be considered a tap (minimum peak height) was set to 20 percent of the maximum peak, do distinguish taps from possible tremor or other disturbances. The minimum interval between each peak was set to 18 samples after examining both healthy signals and signals from Parkinson's disease patients. The minimum peak prominence was set to 15 samples. Figure 3 shows the performance of the peak finding algorithm where it is applied on subject 7 trial 5, which is a patient affected by Parkinson's disease. The displayed signal has been diagnosed with an UPDRS of 1.

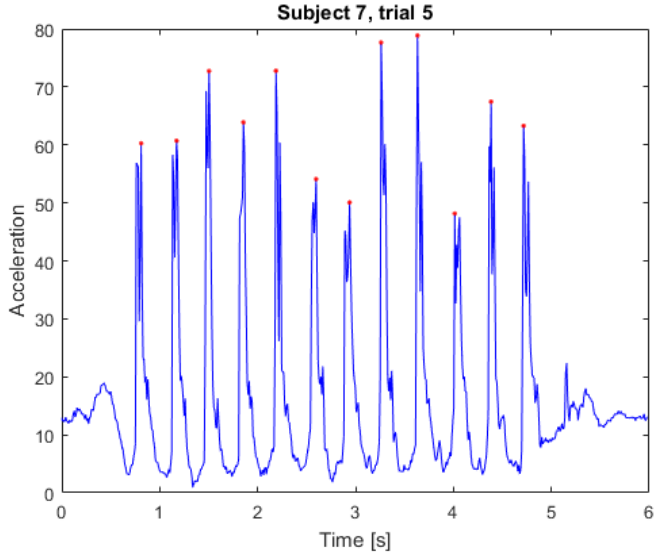


Figure 3: Example of the peak find algorithm for the heel tapping performed, where the red points indicate the coordinates of the found peaks. The algorithm disregards subpeaks in the near proximity of the peaks.

3.3.2 Mean acceleration

Another interesting measure regarding the signals is to measure the mean acceleration. This was done by using the height and number of controlled movements performed as described in section 3.3.1. The height of each peak symbolizing a controlled movement was used to calculate the mean height for each trial. When calculating this feature the fused signal based on the absolute value of all three accelerometer directions was used.

3.3.3 Mean interval time between movements

To find how much difference there is in interval time between a healthy subject and a subject affected by Parkinson's disease, the mean interval time was calculated to measure the time, or number of samples, between each movement. From the function described in section 3.3.1 the locations of the peaks were extracted. The number of samples between each peak was extracted to calculate the mean value of the time between movements for each trial. This feature was used since one of the common treats of bradykinesia is that a patient affected by Parkinson's disease is expected to have a longer interval between the movements, since it might be hard to initiate the movement.

3.3.4 Time variance of maximum acceleration

Another way to generate a feature measuring endurance is to evaluate how the peak amplitude changes over time. Again, the peaks used in section 3.3.1 were used. By using the function `polyfit`, a line was fitted to the extracted points to see if the coefficient of slope is negative, which would indicate that the amplitude decreases over time. This feature might be useful since healthy subjects are expected to have a greater endurance and lesser decrease in performing the task.

3.3.5 Maximum acceleration and angular velocity

As mentioned earlier a patient with bradykinesia makes their movements with lower acceleration and that patients with dyskinesia move with greater acceleration than a healthy subject [21]. Therefore the maximum acceleration and angular velocity might be of interest since for example the maximum acceleration should differ depending on the severity of the symptom of bradykinesia. Dai et. al. proposed using MEMS sensors to record the maximum value for a hand grasping test in order to classify bradykinesia [32]. Griffiths et. al. also used maximum acceleration to detect bradykinesia, but for continuous measurements in the daily life, where the maximum acceleration was calculated for every 2 minutes epoch of the recordings [21]. In order to find the maximum acceleration and the maximum angular velocity the following formula was used:

$$x_{max} = \max(|x[n]|) \tag{9}$$

where n = 1,2,...,N is the sample index in the analysis window and x is the acceleration of the movements or the angular velocity.

3.3.6 Range of acceleration and angular velocity

Another aspect that could be interesting to take into consideration is between which values the acceleration and angular velocities range from. Manson et. al. studied healthy subjects in their study about dyskinesia, to find a healthy range of acceleration. Thereafter the range of acceleration was used as a feature for Parkinson's disease patients, and it was found that patients suffering from dyskinesia often had a wider range of acceleration than healthy subjects [33]. Even though that case was for dyskinesia, this feature might be of interest also for bradykinesia, since theoretically the range should be smaller for patients suffering from bradykinesia due to smaller movements. Equation 10 defines how the range of acceleration and the range of angular velocity was calculated.

$$r_{max} = \max_n(x[n]) - \min_n(x[n]) \tag{10}$$

where n = 1,2,...,N and x is the acceleration of the movements or the angular velocity.

3.3.7 Signal energy

Signal energy has been used as a feature for Parkinson's disease classification in several studies, and Griffiths et.al. stated that the logical conclusion is that patients with bradykinesia should display lower energy than healthy subjects [21]. Bonato et. al. used it while trying to classify Parkinson's disease from signals collected by accelerometers and surface electromyography sensors [8]. Roy et. al. also used Energy as a feature and concluded that it was an important feature in the classification of tremor [34]. The energy of a discrete time signal in is given by the following formula:

$$E = \sum_{n=0}^{N-1} |X[n]|^2. \tag{11}$$

The energy of the signal does increase with the length of the signal and therefore, in order to be able to compare the different measurements, the energy is divided by the length of the signal, i.e. $t_{tot} = \frac{N}{f_s}$ and we get:

$$E_{tot} = \frac{E}{t_{tot}} \tag{12}$$

3.3.8 Signal entropy

In communication theory, entropy is a measure of the efficiency of the system, but in other contexts it might also work as a measure on disorder or uncertainty for a signal or a system. In earlier studies regarding quantification of Parkinson's it has been used as a feature; amongst others Bonato et. al. proposed it as one of the promising features that might be used for the classification of several symptoms in Parkinson's disease [8]. It has also been found that entropy is a good feature when distinguishing between on and off periods for the patient [35]. Tsipouras et al. used the signal entropy to discover dyskinetic movements [11], and Cancela et. al. used it as a feature to detect bradykinesia [9]. Signal entropy is calculated as follows:

$$H(x) = -\int_{x=0}^{\infty} f(x) \log f(x) dx \tag{13}$$

3.3.9 Dominant frequency component

The dominant frequency has been used as a feature in several earlier studies (see [8], [36], [10] and [22]). To calculate this feature, the Fourier transform was applied in accordance with Equation 4. Thereafter the absolute value of the signals was computed and the maximum value was calculated to find the maximum frequency value index k_{dom}:

$$k_{dom} = \arg\max(|X[k]|) \quad for \quad k = 0, ..., N/2 \tag{14}$$

Then to calculate the dominant frequency with respect to the sampling frequency the following equation was applied:

$$f_{dom} = \frac{k_{dom} * f_s}{N} \tag{15}$$

where k_{dom} is the index for the maximum frequency, f_s is the sampling frequency and N is the total number of samples of the signal.

3.3.10 Dominant frequency power

Burkhard et. al. described the dominant frequency energy as an indicator on movement intensities when they used it as a feature to classify dyskinesia [36]. Since bradykinesia is also a movement symptom, this indicates that the Dominant frequency energy might also be a factor to consider during this kind of classification as well. To calculate the dominant frequency energy the power spectral density was used, and the dominant frequency energy is connected to the dominant frequency f_{dom} which was calculated in Section 3.3.9. The frequency energy is described by Rao et. al. [23] based on Parsevals's theorem as

$$E = \frac{1}{N} \sum_{k=0}^{N-1} |X[k]|^2 \tag{16}$$

However, to avoid effects caused by the sampling frequency and dependence on the signal length, in this study a multiplication with the sampling frequency together with a division

of the signal length should be added. This also causes that the calculations will result in the dominant frequency power instead of the energy. Then the formula is adapted in the following manner:

$$E = \frac{f_s}{N^2} \sum_{k=0}^{N-1} |X[k]|^2 \tag{17}$$

For this work, the dominant frequency power was calculated based on the index of the dominant frequency together with the two neighboring frequencies, which is displayed in the final stage of the formula. A consequence of this is the adding of a division by 3 since the sum of the dominant frequency and its neighbors should be averaged.

$$E_{dom} = \frac{2}{3} * \frac{f_s}{N^2} \sum_{k=k_{dom}-1}^{k_{dom}+1} |X[k]|^2 \tag{18}$$

The multiplication by 2 stems from the mirroring characteristics of the DFT and is implemented to use both the positive and the negative frequency components, as used by Wendebourg [22].

3.3.11 Ratio of dominant frequency power to total energy

The ratio of the dominant frequency power to total energy was used as a feature by Patel et. al. when studying classification of different motor symptoms based on accelerometer data. They had found that this kind of feature was one especially suitable to capture signal characteristics from both tremor, dyskinesia and bradykinesia [10]. The total energy of a discrete-time signal is given by Equation 11. Thereafter, the ratio between the dominant frequency power and the total energy is given by

$$r_e = \frac{E_{dom}}{E_f} \tag{19}$$

where E_f is the total energy.

3.3.12 Energy content in three frequency bands

Since different symptoms of Parkinson's disease apply to different frequencies, the energy content in different frequency bands might be an interesting feature. The three different frequency bands was defined based on the main frequency range described in Section 3.3. Based on this they were set as 0.75 - 1.9 Hz, 1.9 - 3.1 Hz, and 3.1 - 4.25 Hz. By rewriting the formula for frequency energy described in Section 3.3.10, the energy for each frequency band was calculated as

$$EC_{b_i^l - b_i^u} = 2\frac{f_s}{N^2} \sum_{k=\frac{b_i^l * N}{f_s}}^{\frac{b_i^u * N}{f_s}} |X[k]|^2 \quad for \quad i = 1, 2, 3 \tag{20}$$

where b_i^l and b_i^u are the boundaries of the frequency band. Gour et. al. used the proportional energy to discover dyskinetic symptoms by observing the energy content in a frequency band ranging between 1.0 - 1.5 Hz [37], and Maetzler et. al. considered energy content in the frequency spectrum to be one of the most promising features for both accelerometer data and gyroscope data [38] for detection of dyskinesia. In the same manner, Tsipouras et.al.

used the energy content in two different frequency bands as a feature [11]. However, the energy content might be of interest also for detection of bradykinesia since that symptom is also expected to have a characteristic frequency representation in the lower frequencies.

3.3.13 Standard deviation of maximum acceleration and angular velocity

One of the features used was the standard deviation of the maximum acceleration and angular velocity, for this Equation 6 in combination with the maximum acceleration/angular velocity was used, this results in the following equation:

$$\sigma^r_{max} = \frac{\sqrt{\frac{1}{M}\sum_{m=1}^{M}\left(x_{max}[m] - \frac{1}{M}\sum_{m=1}^{M} x_{max}[m]\right)^2}}{x_{max}}. \tag{21}$$

where M is the total number of windows. Also, $x_{max}[m]$ is the maximum acceleration/angular velocity for the mth window, and x_{max} is the maximum acceleration/angular velocity calculated for the original, unwindowed signal.

3.3.14 Standard deviation of range of accelerations and angular velocities

With the same reasoning as for Equation 21 the following Equation is retrieved:

$$\sigma^r_{maxRange} = \frac{\sqrt{\frac{1}{M}\sum_{m=1}^{M}\left(r_{max}[m] - \frac{1}{M}\sum_{m=1}^{M} r_{max}[m]\right)^2}}{r_{max}} \tag{22}$$

$\sigma^r_{maxRange}$ is the standard deviation of the range of acceleration and angular velocity, and r_{max} is the range of acceleration/angular velocity calculated for the whole, unwindowed signal.

3.3.15 Standard deviation signal energy

By again considering Equation 6 the standard deviation for the energy of the signal can be calculated:

$$\sigma^r_{energy} = \frac{\sqrt{\frac{1}{M}\sum_{m=1}^{M}\left(E_{tot}[m] - \frac{1}{M}\sum_{m=1}^{M} E_{tot}[m]\right)^2}}{E_{tot}}. \tag{23}$$

σ^r_{energy} is the standard deviation of the signal energy, and E_{tot} is the signal energy calculated for the whole, unwindowed signal.

3.3.16 Standard deviation signal entropy

Once again by considering Equation 6 the standard deviation for the entropy of the signal can be calculated:

$$\sigma^r_{entropy} = \frac{\sqrt{\frac{1}{M}\sum_{m=1}^{M}\left(H[m] - \frac{1}{M}\sum_{m=1}^{M} H[m]\right)^2}}{H} \tag{24}$$

$\sigma^r_{entropy}$ is the standard deviation of the signal entropy, and H is the signal entropy calculated for the whole, unwindowed signal.

3.3.17 Standard deviation dominant frequency

As stated in Section 3.3, the STFT was used to calculate the standard deviation of features derived from the frequency domain. By using the STFT defined in Equation 8, and the formula defined regarding dominant frequency in Section 3.3.9, the estimated dominant frequency for STFT for each interval m is given by

$$\hat{f}_{dom}[m] = \arg \quad \max \left(X_{STFT}[m, f_{STFT}] \right). \tag{25}$$

Then Equation 6 is used to calculate the relative standard deviation, which is given by the following expression.

$$\hat{\sigma}^r_{f_{dom}} = \frac{\sqrt{\frac{1}{M}\sum_{m=1}^{M}\left(\hat{f}_{dom}[m] - \frac{1}{M}\sum_{m=1}^{M}\hat{f}_{dom}[m]\right)^2}}{f_{dom}} \tag{26}$$

Here, f_{dom} is the dominant frequency calculated for the whole, unwindowed signal.

3.3.18 Standard deviation dominant frequency power

To calculate the standard deviation of the dominant frequency power, Equation 18 in Section 3.3.10 should be implemented based on the STFT calculated in Equation 8. This results in the estimated standard deviation of the dominant frequency power when applying Equation 6:

$$\hat{\sigma}^r_{E_{dom}} = \frac{\sqrt{\frac{1}{M}\sum_{m=1}^{M}\left(\hat{E}_{dom}[m] - \frac{1}{M}\sum_{m=1}^{M}\hat{E}_{dom}[m]\right)^2}}{E_{dom}} \tag{27}$$

where E_{dom} is the dominant frequency power calculated for the whole, unwindowed signal.

3.3.19 Standard deviation ratio of dominant frequency power to total energy

In the same manner as the previous section, the estimated standard deviation of the ratio of dominant frequency power to total energy, Equation 19 should be re-calculated with respect to the energy spectrum estimates E_{STFT}. This then gives the estimated ratio $\hat{r_E}$ which is implemented in Equation 6.

$$\hat{\sigma}^r_{r_E} = \frac{\sqrt{\frac{1}{M}\sum_{m=1}^{M}\left(\hat{r}_E[m] - \frac{1}{M}\sum_{m=1}^{M}\hat{r}_E[m]\right)^2}}{r_E} \tag{28}$$

In the same manner as earlier standard deviations, r_E is the ratio calculated for the whole, unwindowed signal.

3.3.20 Standard deviation energy content in three frequency bands

The last standard deviation implemented is the standard deviation for energy content in three frequency bands. Equation 20 is calculated based on the STFT derived in Equation 8 for each window. Thereafter the resulting energy content for each of the three frequency bands should be implemented in Equation 6 to retrieve the standard deviation.

$$\sigma^r_{EC_{b_i^l - b_i^u}} = \frac{\sqrt{\frac{1}{M}\sum_{m=1}^{M}\left(EC_{b_i^l - b_i^u}[m] - \frac{1}{M}\sum_{m=1}^{M} EC_{b_i^l - b_i^u}[m]\right)^2}}{EC_{b_i^l - b_i^u}} \tag{29}$$

Here, $EC_{b_i^l - b_i^u}$ is the energy content for the whole, unwindowed signal.

3.4 Training and validation of methods

There are different approaches used in machine learning to evaluate the trained model and its performance. When a very large amount of observations are available, it may be feasible to form a test set containing a certain percent of the available data directly. However, since the dataset available in this work is quite limited, a cross-validation approach is used to validate the function of each method.

3.4.1 Cross-validation

The k-fold cross-validation approach is based on dividing the dataset into k number of folds with roughly the same size. Then $k-1$ folds are used as a training set and the last fold is used as a test set. By using this approach, MSE might be calculated for each fold, based on the predicted response from the method trained on the $k-1$ training sets. Then this is repeated k times, for every k a new fold is used as a test set. The k-fold CV approximation is calculated by averaging the k estimates of the test error, [16]

$$CV_{(k)} = \frac{1}{k}\sum_{i=1}^{k} MSE_i. \tag{30}$$

The reason for choosing this type of validation is that k-fold cross-validation is suitable for smaller datasets where there might not be a lot of observations available. With a large dataset it is possible to divide the data into a training set and a test set (and possibly also a validation set), and only train once and get sufficient results. Since the dataset used in this work is relatively small, cross-validation is a more suitable approach since the best possible method can be found by training on different test sets for each fold. Figure 4 displays the procedure of cross-validation and how one fold is left out for each iteration and used as a test set.

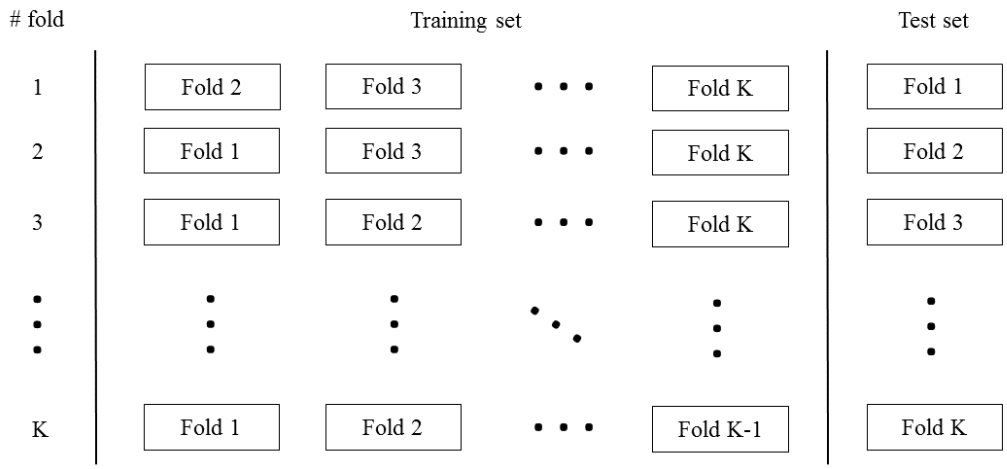


Figure 4: Illustration of the procedure of cross-validation. For each iteration, one fold is left out and used as a test set, whilst the other $k-1$ folds are used combined as a training set.

While implementing the k-folds cross-validation in MATLAB, 18 folds were used for the hand data and the foot data out of the 19 patients in total, due to the fact that one patient lacks hand data and one patient lacks foot data. For the dataset containing hand and foot data combined, 19 folds were implemented. The implementation approach is described below.

1. Divide dataset into k folds
2. for v = 1,...,k;
   (a) Create test set and training set, the test set is the *vth* fold
   (b) Fit model to the training set
   (c) Predict responses of the test set by using the model
   (d) Compute the MSE of the predicted responses
3. Determine the cross-validated MSE according to Equation 30

3.4.2 Nested cross-validation

When using cross-validation to select a model, the same test set is used to choose the tuning parameter and to evaluate the model. This comes with a risk of biasing the model evaluation, therefore one can use nested cross-validation instead. Nested cross-validation uses two different test sets in order to select the tuning parameter and to evaluate the model, which minimizes the bias present in the evaluation. The method consists of an inner and an outer cross-validation. The inner cross validation selects a model and the best tuning parameter, then the outer cross-validation evaluates the model based on the inner cross-validation. The whole procedure is described below:

1. Divide dataset into k folds
2. for v = 1,...,k;
   (a) Create test set and training set, the test set is the vth fold
   (b) for j = 0,...,k-2
      i. Create inner test set and training set, the inner test set is the mth fold, $m = (v + j \bmod k) + 1$, and all the other sets, besides the sets defined as outside test set are defined as inner training set
      ii. For every tuning parameter
         A. Build a model based on the inner training set
         B. Predict responses using model from the previous step and the test set
         C. Calculate the inner MSE
   (c) Calculate the average MSE based on the MSE values calculated for each of the inner folds
   (d) Find best tuning parameters based on the average MSE
   (e) Build a model based on the tuning parameters and the outside training set
   (f) Predict responses based on the previously retrieved model and the outside test set
   (g) Compute the MSE of the predicted responses
3. Calculate the average MSE for all of the folds
4. Compute an approximation of the optimal value for the tuning parameter over all folds

3.4.3 Observation weights

One aspect to consider while training different methods on a dataset is how big the different classes are. If one class is dominating whilst the other ones are much smaller, this might lead to a bias and the model might tend to choose to always assign the output to the dominating class. This is the case for the datasets used in this work and therefore observation weights were implemented for some of the methods. A lot of the model building functions in MATLAB counter this automatically by counting the occurrences of the different classes in the given training set. When this is not done by the function itself, it is often possible to give an additional input to the function, in the form of an observation weight vector. This vector contains how the different classes are weighted inside the overall training data. By this it might be possible to penalize the model harder when it incorrectly classifies an observation from one of the less occurring classes.

3.5 Feature set reduction

There are two ways by which it is possible to reduce the data into a suitable feature set; feature selection and feature extraction. Feature selection involves selecting a number of features from the original feature set that are deemed suitable and informative for that kind of signal. This means that the selected features are not changed in any way, but rather that the rest of the features that are deemed not interesting are discarded. Feature extraction, on the other hand, involves transforming the feature set by different available methods, such

as Principal Component Analysis (PCA) for example, to create a new, reduced feature set with a smaller number of principal components based on the original feature set [14]. Both these approaches will be evaluated in this work.

3.5.1 Normalization of features

Some machine learning methods might be sensitive to large differences in the feature set. This might occur when the dynamic range of the different features have large variance, due to the fact that the features might represent a large variety of aspects of the data. It might also be inflicted by features having different units of measurement. To improve the accuracy and prepare the feature set before building methods based on it, the features used should therefore be normalized [14]. For this work, the feature vectors were normalized with respect to the mean and the standard deviation of each feature vector in the following manner:

$$x_{i,norm} = \frac{x_i - \bar{x}}{x_{std}}, \quad for \quad i = 1, ..., N \tag{31}$$

where x_i is the ith feature vector of the N total feature vectors. The mean and the standard deviation of the feature vector is $\bar{x}$ and x_{std}, respectively.

3.5.2 Outlier detection

To further improve the pre-processing of the feature set, a method for detecting outliers had to be applied. Otherwise these outliers might affect the outcome of the learning methods negatively, and therefore the effect of them should be minimized. When working with outlier detection, a threshold has to be determined to decide on a suitable range for the feature vectors. One standard procedure when choosing a suitable threshold is to multiply the standard deviation of the feature vector with an integer, where a distance of 3 times the standard deviation covers about 99 percent of the points for a normally distributed random variable [14]. By using this, diverging values could be detected in each feature vector. However, this method might be flawed when the feature set contains large outliers, since both the mean and the standard deviation is largely affected by outliers. Therefore for this work, median absolute deviation (MAD) was adapted instead, which uses the median of the feature vector which is not affected by a potentially large outlier. The median is only negatively affected if more than 50 percent of the given vector are outliers or goes to infinity. The equation for MAD is defined as follows [39]:

$$MAD = b * M_i(|x_i - M_j(x_i)|) \quad . \tag{32}$$

In this equation, $M_j(x_i)$ is the median value for the ith feature vector. M_i is the median value after subtracting the feature vector with M_j and taking the absolute value. The sensitivity factor, b, should be determined to retrieve a suiting threshold for the outlier detection [39]. While implementing this method for outlier detection, the found outliers in the feature set were exchanged for NaN for further processing. There are several proposed methods on how to deal with the missing data obtained when removing outliers. One procedure, which was also implemented for this work, is to substitute the missing data by a nearest neighbors approach [40]. The nearest neighbors in this context is the closest non-missing values in the vector containing the same kind of feature. To solve this in MATLAB the built-in function `fillmissing` was used, which allows the choice of what kind of method that should be applied when replacing the missing values. In this work, the method `'linear'` was used,

which applies linear interpolation between the nearest non-missing values, e.g. the two values of the same kind of feature, and adds it to the missing element. This means that if for example observation 10 is missing a value for feature 2, the interpolation is done based on the value for feature 2 from observation 9 and 11. Besides only using the procedure described, the feature set was also analyzed visually to detect values diverging greatly.

3.5.3 Forward selection

Forward selection is a kind of stepwise regression method, which is used to select a subset of the original feature set to build a model. It works by adding one feature at a time to the model and examining if that particular feature improved the model. It is a greedy approach since it does not necessarily find the absolute best model out of all possible 2^p (where p is the number of features) models, since it does not take a step back in the end to evaluate if any features that was added early has become redundant and does not contribute to the final model anymore. Forward selection may be a good solution for feature selection when a large feature set is available [16]. However, it might also be computationally expensive, since it involves examining a large variety of candidate feature sets.

Forward selection is used by first building a null model which only contains the intercept parameter. Then nested cross-validation is used to evaluate all features and the one that improves the model the most is added. Next step is to evaluate all the remaining features, again adding the best one, and so on [14]. This means that the number of features works as a tuning parameter which could be any value between zero and the maximum number of features. To determine which feature is the best one for each iteration, the sum of squared residuals (RSS) was used, which should be minimized. The formula for RSS is [16]:

$$RSS = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{33}$$

Here, y_i is the true UPDRS value and $\hat{y}_i$ is the predicted UPDRS value. Nested cross validation was used with forward selection to determine the best number of features to use for the best fitted model. In this work, forward selection was implemented together with linear regression. The implementation is presented in pseudo-code below.

1. Prepare for cross-validation by extracting a test set and a training set for each outer fold
2. Prepare for nested cross-validation by extracting a test set and a training set for each inner fold
3. for $i = 1, ...,$ Number of folds: Use forward selection together with linear regression for each fold
   (a) for $j = 1, ...,$ Number of inner folds; Use forward selection together with linear regression for each inner fold
   (b) Create a null model, which contains only the intercept parameter
   (c) Predict responses by applying the null model on the jth inner fold test set
   (d) Calculate MSE for the jth model variant
   (e) for $k = 1, ...,$ Max number of features; Create models for each number of features

i. for $l = 1, ...,$ Initial number of features; Build and evaluate all models containing k features
   A. Build model with l number of features
   B. Save the variable name(s) of the l number of features
   C. Predict responses by applying the built model
   D. Calculate RSS for the current model
ii. Extract the index of the minimum RSS value to decide which added feature that improved the model the most
iii. Build a new model with the best model variable(s) to get the best current model
iv. Predict responses by applying the best current model
v. Calculate MSE based on the best current model
vi. Update next set of features to consider by removing the best previous feature before the next round of the training part
vii. Update initial number of features to consider

4. Calculate the average MSE value from the MSE values for all the inner folds
5. Extract the minimum of the average MSE values to decide the optimal number of features
6. for $j = 1, ...,$ Optimal number of features; Build and evaluate models containing j features
   (a) for $k = 1, ...,$ Max number of features; Build and improve the final model
      i. Build the model with k features
      ii. Save the variable name(s) of the k number of features
      iii. Predict responses by applying the model
      iv. Calculate RSS for the model
   (b) Decide again which of the added predictors that improved the model the most
   (c) Update next set of predictors to consider by removing the best previous predictor before the next round of the training part
   (d) Update remaining part of features to consider
7. Evaluate the best fitted model by predicting responses on the test set
8. Calculate MSE for each outer fold

3.5.4 Principal component analysis

Principal component analysis is a feature extraction method, which results in a subset of the original feature set. PCA can be a very powerful tool to use with methods that are sensitive to higher dimensions, since only a few principal components can be used if desired. PCA is used to find the vector in the feature space containing the most variance in the data, which will be named the first principal component. That vector is also the one found with the shortest distance to all data points given. The second principal component is always perpendicular to the first one, and contains the greatest variance for the points uncorrelated

to the first principal component. When building the vector for a principal component, each element in the vector is given a positive or negative score displaying the distance to the current data point [16].

For this work, PCA was used as a dimensionality reduction tool together with some of the classification and regression methods. PCA was implemented using the built-in MATLAB function `pca`, which requires the training set as input. The additional setting "Centered" was set to `'false'`, since this means centering the data points by subtracting the means, which is not needed since the data has already been normalized in the pre-processing stage described in Section 3.5.1. The mentioned function may be used with different kinds of algorithms, but when implementing this the default algorithm, which is singular value decomposition (SVD) was used. For methods that do not require any tuning of some parameters, nested cross-validation may be used to decide on an optimum number of principal components to use while training the data. This was the approach used when implementing PCA together with linear regression. In other cases where nested cross-validation is already used for parameter tuning, it might not be suitable to have PCA as another parameter to be decided. One reason is that it becomes more computationally expensive since it would largely increase the number of iterations needed to explore all possible alternatives. In those cases were tuning of a parameter was already implemented, a fixed number of principal components was used instead. This number should be decided based on the kind of classification and regression methods for which PCA should be implemented. One way to decide how many principal components to use, is to consider the percentage of variance explained for each principal component. Taking this into consideration, and the fact that the implementation of smoothing splines can only handle two dimensions, it was decided that the two first principal components should be used for further implementation. The variance can be extracted within the function `pca` by adding the voluntary output `'explained'`. As an example, the variance for the first principal component when applied to all features from the foot data from Parkinson's disease patient is 43 percent, whilst the variance of the second principal component is 21 percent. The third one has a lower variance of 5 percent which makes it clear that the first two PC's are the ones containing the most variance.

3.6 Machine learning methods

For this work, a selection of classification and regression methods are used to predict the UPDRS rating based on the features for all signals. Even though some of the methods used are regression methods, the predicted responses when validating the method were rounded to the nearest integers, to be able to compare them with the true responses of the UPDRS scale, which are as mentioned expressed in integers ranging between 0 and 4, or in this implementation 0 to 3 due to the lack of signals diagnosed with UPDRS 4 according to the physicians.

To be able to present a fair comparison between the different methods used, the mean squared error (MSE) was used within all models. It was implemented after the training phase of each model, whereas the trained model was evaluated by letting it predict responses (UPDRS values) to test data. By using these predicted responses and the true responses known for the same data, the MSE value was calculated as [16]:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y_i})^2 \tag{34}$$

n is length of UPDRS vector, y_i is the true UPDRS, $\hat{y_i}$ is the estimated UPDRS from the

model. If the model has achieved to predict responses with values near the true responses, the MSE value will be small. For this work, the methods implemented were forward selection, linear regression, ridge regression, the lasso, local regression, smoothing splines, multivariate adaptive regression splines, K-nearest neighbors, decision tree, neural network, and support vector machine. Principal component analysis was implemented together with six of these methods, namely decision tree, K-nearest neighbor, linear regression, local regression, smoothing splines, and neural network. The implementation in MATLAB is quite similar between some of the methods used, with differences regarding the tuning parameters and functions used. Shown below is pseudo code for a generic implementation. Differences between the different methods are described within each subsection describing the methods.

1. Prepare for cross-validation by extracting a test set and a training set for each outer fold
2. Prepare for nested cross-validation by extracting a test set and a training set for each inner fold
3. for $i = 1, ...,$Number of folds
   (a) for $j = 1, ...,$Number of nested folds
      i. Generate a range of values for the tuning parameter
      ii. Build a model based on the jth training set and for each possible tuning parameter value
      iii. Predict responses by evaluating the built model
      iv. Calculate MSE for each model variant with each possible number of neighbors
   (b) Calculate the average MSE based on the MSE values calculated for each of the nested folds
   (c) Find the optimum tuning parameter value for each nested fold by finding the minimum value of the average MSE values
   (d) Build a model based on the ith training set and for all the optimum tuning parameter values from the nested folds
   (e) Predict responses and calculate the MSE for each outer fold
4. Calculate the average MSE based on the MSE values calculated for each of the outer folds
5. Calculate the average optimal tuning parameter based on the optimal tuning parameters for each of the outer folds

3.6.1 Linear Regression

Linear regression is a quite simple model based on an assumption that one may find quite a linear relationship between an observation X and a response Y, and uses a least squares approach to fit the model, which minimizes the sum of squared residuals. The formula for linear regression is quite straightforward, and is mentioned by James et. al. as [16]

$$\hat{Y} = \beta_0 + \sum_{n=1}^{N} \beta_n x_{in} \tag{35}$$

where β_0 is the first model coefficient, called the intercept term, and N is the number of features (not to be confused with the earlier notation where N was the number of samples). A feature is described by x_{in} where i denotes the current observation.

When implementing linear regression in MATLAB, the function `fitlm` was used to build the model, and `feval` was used to evaluate and predict responses of the UPDRS rating. Linear regression does not require any nested cross-validation since tuning of parameters is needed. Therefore, the implementation is done with cross-validation only described by the following pseudo code:

1. Prepare for cross-validation by extracting a test set and a training set for each fold
2. for $i = 1, ...,$Number of folds; Apply linear regression to each fold
   (a) Build a linear regression model for the ith training set
   (b) Evaluate the model by predicting responses for the ith test set
   (c) Calculate MSE for the ith fold
3. Calculate the average MSE based on the MSE for each fold

3.6.2 Ridge Regression

Ridge regression works in a similar way as a least squares fitting described by James et.al., with the difference that the coefficients are estimated by estimating a somewhat different quantity[16]. The ridge regression coefficient approximation $\hat{\beta^R}$ minimize Equation 36

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 = RSS + \lambda\sum_{j=1}^{p}\beta_j^2, \tag{36}$$

the scaling λ is a so called tuning parameter. It has to be larger than or equal to zero. Ridge regression is trying to, by making the RSS small, find coefficient approximations that fit the data well. It is said to be fitting for high-dimensional datasets as well as sets that are believed to have large collinearity, meaning that features in the model are highly correlated [16]. The shrinkage penalty, $\lambda\sum_{j=1}^{p}\beta_j^2$, shrinks the estimates of β_j towards zero, since it is small when $\beta_0, \beta_1, ...\beta_p$ are almost zero. To control the effect on the ridge regression coefficients approximation from the two terms mentioned, λ is used. If λ is moving towards infinity, β moves towards zero. If on the other hand λ is equal to zero, the penalty term will have no effect and ridge regression will be the same as least squares[16]. For each value of λ, ridge regression will return a new set of coefficient approximations, $\hat{\beta^R}$. The range of possible λs could range between zero and infinity. For this implementation, values for λ between 0 and 5000 was investigated by the nested cross-validation, with a step size of 10, to cover a large range but still keep the number of iterations at a reasonable level.

The method was implemented in MATLAB using the built in function `ridge` which takes a vector with predictions, the training dataset and a vector with tuning parameters (λ). The implementation follows the same procedure as the generic pseudo code presented in the beginning of Section 3.6.

3.6.3 The Lasso

Ridge regression as described in Section 3.6.2 builds a model with all features generated, which can be a challenge when it comes to interpretation if the number of features is large, even though all the features might not be important. Ridge regression will return a model that contains all features instead of evaluating which are the most important ones and building the model with regards to them. As an alternative to ridge regression one can use the lasso which performs variable selection. This will result in a model that is much easier to interpret than ridge regression [16]. The lasso coefficients, $\hat{\beta}_L^2$, minimizes Equation 37

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j| = RSS + \lambda\sum_{j=1}^{p}|\beta_j|. \tag{37}$$

The difference between ridge regression and the lasso is that the term β_j^2 is exchanged by $|\beta_j|$ in the lasso. It is said that the lasso generates sparse models, i.e. models that involve only a subset of variables [16].

The implementation of the lasso in MATLAB resembles the implementation of local regression, with the difference that instead of using `ridge`, the built in function `lasso` was used. It also takes the vector with predictions, the training dataset and two arguments: alpha and Lambda as inputs, where alpha is a value between 0 and 1, if the value is set to 1 (as in this project) it represents lasso regression, if it is close to 0 it represents ridge regression. Lambda is a vector with tuning parameters, which was implemented investigating values between 0 and 0.5, with a step size of 0.01. This is loosely based on an implementation example presented by James et. al. [16] where a range of lambdas between 0.005 and 5 were tested. However, their results gave that the cross-validation error increased greatly after 0.5, therefore this upper limit was used. Lambdas of about the same size were also used by Weisberg [41], who used cross-validation to find reasonable values, yielding lowest lambda value to be 0.004.

3.6.4 Local Regression

When it comes to fitting a flexible non-linear function, local regression is a common approach. With local regression the fit of the target point is computed only by using the nearby training observations, and allows overlap in the classification regions [16].

Given a set of training points whose x_i are closest to x_0, a weight, $K_{i0} = K(x_i, x_0)$, is assigned to every point in the neighbourhood. The closest point is assigned the highest weight and the point furthest away from x_0 has the lowest weight. Then Equation 38 is minimized by finding β_0 and β_1,

$$\sum_{i=1}^{n} K_{i0}(y_i - \beta_0 - \beta_1 x_i)^2 \tag{38}$$

Then the fitted value at x_0 is given by

$$\hat{f}(x_0) = \hat{\beta}_0 + \hat{\beta}_1 x_0. \tag{39}$$

The built-in MATLAB function is only able to handle two features at the time. Therefore the toolbox Locally Weighted Polynomials made by Jekabsons [42] is used which is able to build a model for the whole feature set. This method requires a set of parameters, and

for this the function `lwpparams` is used, where the input arguments are Kernel type, if KNN should be used, and the window size applied. The range of the tuning parameter to be tested was chosen as 0.4 to 1, with a step size of 0.05. This range was loosely based on values implemented by James et.al. [16], who used parameter values of 0.2 and 0.7 when building a local regression model. The tuning parameter controls the rigidness of the decision boundaries, whereas a smaller value fits the model more closely to the data [16]. The toolbox offers a function to build the model, which is called `lwppredict`, which both builds a model based on a training set and outputs predictions based on a test set. It also takes as input the parameters decided by `lwpparams`. The implementation follows the steps explained by the pseudo code presented in the beginning of Section 3.6.

3.6.5 Smoothing splines

When fitting a smooth curve to a set of data, the resulting $RSS = \sum_{i=1}^{n}(y_i - g(x_i))^2$ should be small. If no constraints are put on $g(x_i)$, RSS can be equal to zero by choosing g so that it interpolates all of the y_i. This type of function would be too flexible and overfit the data. What is desirable is a function that makes RSS small, but is smooth at the same time [16]. To affirm that g is smooth, we want to find the function g that minimizes

$$\sum_{i=1}^{n}(y_i - g(x_i))^2 + \lambda \int g''(t)^2 dt, \tag{40}$$

where λ is the tuning parameter and the function g is the smoothing spline. For g to be smooth, $g'(t)$ is nearly constant and $g''(t)$ is small, also the larger the value of λ the smoother g will be. If λ is close to infinity, g will just be a line that goes through the training points as close as possible. In that case, g will be a linear least squares line. The parameter λ affects the effective degrees of freedom and therefore also the smoothness of the method [16]. For this implementation, values between 0 and 0.2 with a step range of 0.001 was investigated. A suitable range between 0 and 1 has been proposed [43], but for this implementation lower values gave a better result which resulted in the lower range.

The function `tpaps` in MATLAB is used to construct a thin-plate smoothing spline. The inputs to the function are the vector with predictions, the training dataset and tuning parameter λ. Then the model built from `tpaps` is evaluated with the function `fnval`. This implementation also resembles the generic pseudo-code described in Section 3.6.

3.6.6 Multivariate adaptive regression splines

The main purpose of multivariate adaptive regression splines (MARS) is to predict a continuous dependent variable from a set of independent variables. MARS builds a relationship between the dependent and the independent variables from a set of coefficients and basis functions that are driven from the regression data. The method operates as a multiple piecewise linear regression, "the region of application" for a specific linear regression equation is defined by each breakpoint [44]. To estimate the relationship between the response and the features, MARS applies two-sided truncated basis functions of the form (Equation 41) as basis functions for linear or nonlinear expansions.

$$h_m = (x - t)_+ = \begin{cases} x - t, & \text{if } x > t \\ 0, & \text{otherwise} \end{cases} \tag{41}$$

where t is the knot of the basis function, the knots are determined from the data. To denote that only positive results are considered there is a plus sign next to the term. The general equation for MARS is given by

$$f(X) = \beta_0 + \sum_{m=1}^{M} \beta_m h_m(X), \tag{42}$$

where β_0 is the intercept parameter and the sum of one or more basis function $h_m(X)$, which is weighted by β_m [44].

To implement MARS the three functions `aresparams`, `aresbuild` and `arespredict` from the toolbox Adaptive Regression Splines by Jekabsons [45] was used since there are no built-in functions for MARS in MATLAB. In the inputs of `aresparams` it is defined if the model should be piecewise-linear or cubic-linear, in this project both approaches were tested. Also the number of basis functions are defined and the function returns a structure of training coefficients. Then `aresbuild` is used and it takes the vector with predictions, the training dataset and the parameters retrieved by `aresparams`. At the last step `arespredict` is used to predict the UPDRS of the patients. The number of basis functions defined is the tuning parameter that should be decided by nested cross-validation. The range that was tested was 1 to 20 basis functions. However, due to this method being extremely computationally expensive in time when implemented with nested cross-validation, this was only completed once to find the average optimum number of basis functions. During that time, the implementation was done in the same manner as the pseudo code described in Section 3.6, and for the other runs it was used without nested cross-validation and the value of the optimum tuning parameter fixed. This was decided to be 15.4444 and 11.898 for MARS linear and MARS cubic respectively, which was rounded to 15 and 12 since the model function requires the number of basis functions to be an integer.

3.6.7 K-nearest neighbors

K-nearest neighbor bases its classification on the highest probability for a certain object to belong to a certain class, by estimating the distribution of classes and using Bayes rule to classify based on which class that is more represented. This is implemented by determining which objects are in near proximity (determine k neighbors) from the object that it is attempting to classify [16]. James et. al. presents how the K-nearest neighbor method estimates the conditional probability for a certain class j [16]:

$$P(Y = j|X = x_0) = \frac{1}{K} \sum_{i \in N_0} I(y_i = j) \tag{43}$$

where N_0 is the representation of K points that are the closest neighbors to the specific object of interest, x_0. The classes used in this project is the available UPDRS ratings, e.g j is equal to 0, 1, 2, 3, or 4. When using K-nearest neighbor, it's important to choose a suiting number of nearest neighbors k to consider, since this greatly affects the outcome of the method. By setting k too small or k too high the method becomes too flexible or too rigid, respectively [16].

Nested cross-validation was used to determine the best number of neighbors to use, e.g. the optimum tuning parameter. The range to be tested for the number of neighbors to use was 1 to 50. The number of possible neighbors should be some number below the number of observations available, and it affects the flexibility of the decision boundary. A low value

for the number of neighbors cause a very flexible decision boundary, while it gets more rigid with a larger value [16]. To build a model for K-nearest neighbor, the MATLAB function `fitcknn` was used, which builds the model based on a fixed set of the k nearest neighbors. To test the model and predict the UPDRS values the MATLAB function `predict` was implemented. As stated earlier the mean squared error (MSE) was calculated to get a clear view of the reliability of the method, by implementing Equation 34.

3.6.8 Decision trees

Decision trees are methods using a set of splitting rules to stratify, or segment, the predictor set used. To make this kind of method able to compete with some other supervised methods with high accuracy, an approach where several model trees are produced are often used. These variants are later combined into one model for the final prediction. The splitting takes place in nodes in the tree, which makes up the branching of the tree [16]. There are several different methods available on how the decision tree should decide on how the best model is built. In this work, recursive binary splitting, which is a greedy approach, is applied. It is presented by James et. al. [16] and is considered greedy since it does not explore all possible subsets of the feature space. Rather it uses RSS to check which subset division that gives the largest decrease in RSS, not looking ahead or re-evaluating earlier branches. To further increase the accuracy of a decision tree, a technique called pruning is often applied, which removes branches that do not affect the decrease the RSS value. To achieve this, a tuning parameter is used together with a minimization criteria. Pruning also helps to avoid overfitting, since the removal of some branches decreases the complexity of the tree, and therefore the tree becomes less specific with regards to the training data [16].

When implementing the decision tree method, the MATLAB function `fitctree` was used to build the model, which requires a training set with features as well as labels (UPDRS values). The function also offers the option to active a "Prune" alternative, which has the consequence that the function estimates the optimal sequence of pruned subtrees. MATLAB also has a built in function for the pruning itself, called `prune`, which was used after building the model. The method was implemented together with nested cross-validation, where the inner cross-validation was used to find the optimal tuning parameter α, which is used by the pruning function to decide on the level of pruning. If the parameter is set to zero, no pruning is done, and the level of pruning increases with an increasing parameter value. However, if the parameter value is too large the tree will be pruned to its roots. Pruning is done to remove subtrees that does not improve the tree in any way. The function in MATLAB evaluates the weight given to each node in the tree and eliminates nodes with zero weight or weight below a certain threshold. The range of tuning parameters to be tested was 0 to 0.05 with a step size of 0.0025. In general for this work, a tuning parameter value above 0.05 resulted in the tree being pruned to its roots, therefore that value was set as the upper limit. As for the implementation, Decision tree was used as described in the pseudo-code of Section 3.6 with the exception that after step 3.a.ii and 3.d, the pruning function was implemented before evaluating the model on the test set.

3.6.9 Neural networks

Neural networks constitute a large variety of algorithms used within both the classification and the regression approach. The name and approach is inspired by the neurons in the brain, which are activated when new information is received and sent further in the network. A

neural network is made up by layers and nodes (neurons), where the nodes between each layer are interconnected to fit an input to a desired output [46].

While implementing a neural network, the input layer consists of a number of nodes equal to the number of features. Some variants however also implement an additional node as a bias term. For the output layer it is most common to have only one node since it is mostly desired to have just one response to a certain observation. Between the input and output layer one or more hidden layers may be implemented. For a lot of cases only one hidden layer is enough, and adding more hidden layers might not improve the performance significantly but rather only increase the computational cost of the algorithm [46]. The preferred number of nodes to use for the hidden layer might vary, but it is standard to a number between the number of nodes used in the output and the input layer.

When implementing neural networks in MATLAB, a feed-forward network using the Levenberg-Marquardt algorithm was used, which is implemented by using a function called `fitnet`. This function builds a two-layer network with one hidden layer. To train the network on a given training set, the function `train` was used. In this work, nested cross-validation was used to find the optimum number of nodes to use in the hidden layer. The range of possible values was between 1 and the maximum number of features. That number of nodes were later used directly when running different trials, since it was proven to be very computationally expensive to use nested cross-validation for each run considering neural networks. This number of nodes, which was calculated as the average of the best number for each of the outer folds, where chosen to be 44 nodes for the inner layer. However, this number was reduced for the datasets containing less than 44 features to keep the number of nodes in the hidden layer at a value between the number of inputs and the number of outputs. The implementation of neural networks is presented below in pseudo-code, here without nested cross-validation since this was only used initially to retrieve a suiting number of nodes to use further on. In the same manner as for decision trees, it is possible to use pruning while building a neural network model, which may be used to remove unnecessary neurons that does not contribute to the final performance of the network. Pruning was implemented by using the function `prune`. Since the function was only used with nested cross-validation once, the implementation resembles the implementation of linear regression in Section 3.6.1, with the difference that the pruning function is implemented after step 2.a.

3.6.10 Support vector machine

The support vector machine (SVM) is an extension of the support vector classifier, it is a result from making the feature space larger by using kernels. A support vector machine converts a linear classifier into one that creates non-linear decision boundaries in an automatic way [16]. If a boundary between two classes is linear, the support vector classifier is a natural choice in the two-class setting. Sometimes the class boundaries are non-linear and then the support vector classifier will not work. When this occurs, making the feature space larger by using functions of the features, such as quadratic and cubic terms, or even higher order polynomials, could be implemented to implement a non-linear decision boundary [16]. Rather than fitting a support vector classifier using p features, $2p$ features could be used for instance, as in the quadratic polynomial case. A support vector classifier builds the decision boundary by forming a hyperplane. By using the optimization criterion given in Equation 44 the hyperplane is given.

$$\text{maximize M} \qquad \text{subject to } y_i \left(\beta_0 + \sum_{j=1}^{p} \sum_{k=1}^{d} \beta_{jk} x_{ij}^k \right) \geq M(1 - \epsilon_i), \tag{44}$$

$$\sum_{i=1}^{n} \epsilon_i \leq C, \quad \epsilon_i \leq 0, \quad \sum_{j=1}^{p} \sum_{k=1}^{d} \beta_{jk}^2 = 1,$$

M is the width of the margin, this quantity should be as large as possible. C is a positive parameter deciding the severity of objects being on the wrong side of the margin, and $\epsilon_1, ..., \epsilon_n$ are slack variables that allow individual observations to be on the incorrect side of the hyperplane. To still keep the method computationally feasible, since an expansion of the feature space increases the complexity, a support vector machine handles the observations by letting the method use only the generalized inner product of different observations pair [16].

The linear support vector classifier can be represented as

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle \tag{45}$$

To approximate the function $f(x)$, the inner product between the new point x and each of the training points x_i has to be calculated. If a training observation is not a support vector, then α_i equals zero. If $\mathcal{S}$ is the collection of indices of these support points, Equation 45 can be rewritten as:

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i \langle x, x_i \rangle \tag{46}$$

If the inner products are then expanded, one can see that $f(x)$ is a linear function of the coordinates of x. It also establishes the symmetry of α_i and the original parameters β_j [16].

The MATLAB function `templateSVM` was used to get a support vector machine learner template with parameters suitable for another function called `fitcecoc`, which is the function building the model. The tuning parameter range to be tested was between 0.01 and 0.2, with a step size of 0.01. This range was chosen based on implementation examples done by James et. al, where parameter values of 0.001, 0.01 and 0.1 were used. The tuning parameter is used as input to the `templateSVM` function. The value affects the flexibility of the decision boundary and how hard faulty classifications should be penalized [16]. As for several other methods, SVM was also implemented according to the pseudo code described in Section 3.6. For this work, SVM was used with three different kernels; linear, cubic and polynomial.

3.7 Hypotheses for feature set

Before implementing the methods on the data, some hypotheses were formed with different test cases, to be able to determine if a certain kind of features affected the performance more than others. This also allows the methods to be tried on different sizes of feature sets, since some methods may have a problem with feature sets of higher dimensionality. Testing of different hypotheses might also be of value for future studies and implementation, since it might indicate what kind of sensors that are most effective in this kind of diagnostics. These formed hypotheses was used in all the methods described in Section 3.6. For the methods

where PCA was implemented as a feature extraction step, the principal components were based on the features from each hypothesis. For smoothing splines, which can only handle two features with the kind of implementation used in this work, forward and backward selection was used to find the features which improved the model the most. These two features was then implemented with the smoothing splines method. All hypotheses will be evaluated for foot data, hand data, and foot and hand data together. Besides evaluating data from Parkinson's disease patients with UPDRS rating between zero and three, the methods will also be used to see if it is possible to judge between UPDRS of zero from healthy controls and UPDRS of zero from Parkinson's disease patients. This could be of interest since a positive result might indicate that machine learning algorithms are able to detect Parkinson's disease characteristics which cannot be detected by visual inspection from a clinician. This will however only be evaluated for foot data, due to the lack of a sufficient amount of hand data signals from Parkinson's disease patients diagnosed with UPDRS 0. The formulated hypotheses are presented below.

3.7.1 Hypothesis 1: All available features

The first hypothesis is to use all the available features, which are 164 features generated according to the described procedure in Section 3.3. This means that features generated from both the accelerometer and the gyroscope data was used, from the x-, y-, and z-direction as well as the vector magnitude according to Equation 1. The division in all the 164 features are 80 features from the gyroscope data, and 80 features from the accelerometer data. The last 4 features are the visual features described first in Section 3.3, which were generated from the XYZ data from the accelerometer.

3.7.2 Hypothesis 2: All features from the accelerometer data

To further explore the effect of the accelerometer data, the second hypothesis uses the signals from all directions gathered by the accelerometer. By using all three directions, but only for one sensor, some other aspects could be covered. One aspect could be the possibility that a Parkinson's disease patient might move the leg more in the horizontal plane than a healthy subject which might be able to control the movement more and execute it more focused tapping on the same spot. A total of 80 features are captured within this dataset.

3.7.3 Hypothesis 3: All features in the y-direction from the accelerometer data

Another hypothesis that might be interesting is to use the accelerometer features only from the y-direction, since the y-direction was the direction of the sensor which pointed down vertically alongside the leg whilst executing the movement. Therefore the features extracted from the y-direction could have a greater influence on the classification than the other directions. This subset contains 20 features.

3.7.4 Hypothesis 4: All features from the gyroscope data

The next hypothesis uses all the features gathered from the gyroscope signals in all directions. Intuitively, this dataset should work better for the hand data, considering the turning movement used for the hand exercise, which should be easier to pick up for the gyroscope than the accelerometer. As for hypothesis 2, this dataset contains 80 features.

3.7.5 Hypothesis 5: Magnitude features (XYZ) from the accelerometer data

For the kind of movement test used to gather the data for this work, the theory is that the accelerometer data should have a greater influence on classification than the gyroscope data, since the movement is a vertical movement without any turning of the foot. Therefore the next hypothesis is based only on a subset of features, namely the ones from the accelerometer data, but only the merged data from all three directions is used to explore the effect the absolute value of all three directions may have on the classification. This subset contains 24 features.

4 Results

In the following section the results obtained in the project are presented. The numeric results are presented in Table 2, Table 3, Table 4 and Table 5. For each case, the average cross-validated MSE are introduced for all hypotheses. The best and the worst average cross-validated MSE and its corresponding method are indicated by green or red in each table. This has been marked for all five hypotheses. More detailed tables for each Hypothesis, together with the standard error of both the MSE and the tuning parameter are presented in the Appendix. The average best tuning parameter is presented for each model (if applicable) below the MSE in a parenthesis. The meaning of each presented tuning parameter is presented for each method described in Section 3.5 and Section 3.6. Throughout this section, first the overall results for each case is presented. After that, more detailed results are displayed for each of the most promising models for each Hypothesis.

4.1 Foot Data: patients rated with UPDRS 0 - UPDRS 3

The first dataset used was the one with data collected from sensors that were attached to the patients ankles for patients with UPDRS ratings between 0 and 3. As described in Section 3.7, the conjecture is that due to the characteristic of the heel tapping exercise, accelerometer data in the y-direction is expected to be the most important part of the available signals. Table 2 displays the results obtained by all the models applied, where the average cross-validated MSE value is displayed, together with (if applicable) the related average cross-validated tuning parameter.

Method	**Hyp 1**	**Hyp 2**	**Hyp 3**	**Hyp 4**	**Hyp 5**
Forward selection	0.863	0.986	0.773	0.791	0.810
	(41.111)	(19.222)	(8.056)	(32.667)	(9.111)
Decision tree	1.377	1.348	1.046	1.071	1.233
	(0.043)	(0.030)	(0.022)	(0.031)	(0.021)
K-nearest neighbor	0.777	0.997	0.873	0.973	0.820
	(15.056)	(15.722)	(10.611)	(40.611)	(8.833)
Linear regression	2.781	2.626	0.800	0.689	0.798
Local regression	2.632	3.283	0.918	0.683	0.790
	(0.678)	(0.672)	(0.669)	(0.700)	(0.650)
Smoothing splines FS	0.749	0.673	0.673	0.631	0.707
	(0.150)	(0.099)	(0.099)	(0.153)	(0.098)
MARS linear	1.194	1.069	0.953	1.076	0.737
MARS cubic	1.103	0.891	0.634	0.868	0.739
Neural networks	1.703	1.614	1.560	1.363	1.374
Ridge regression	0.661	0.777	0.706	0.617	0.740
	(37.778)	(70.556)	(26.111)	(13.333)	(18.889)
The lasso	0.819	0.776	0.755	0.648	0.697
	(0.012)	(0.012)	(0.016)	(0.004)	(0.017)
SVM cubic	0.834	0.842	0.917	1.014	0.813
	(0.010)	(0.011)	(0.137)	(0.014)	(0.159)
SVM poly	0.817	0.804	0.991	1.066	0.889
	(0.010)	(0.010)	(0.104)	(0.010)	(0.057)
SVM linear	0.771	0.792	0.790	0.903	0.604
	(0.128)	(0.171)	(0.159)	(0.157)	(0.128)
PCA decision tree	1.088	0.895	0.867	0.881	1.022
	(0.096)	(0.046)	(0.101)	(0.046)	(0.063)
PCA K-nearest neighbor	0.962	0.854	1.052	1.032	0.922
	(38.111)	(40.389)	(39.167)	(44.222)	(29.000)
PCA linear regression	0.442	0.8983	0.750	0.697	0.668
	(15.111)	(17.111)	(8.056)	(65.167)	(8.222)
PCA local regression	0.964	0.949	0.954	0.995	0.922
	(0.414)	(0.833)	(0.675)	(0.594)	(0.844)
PCA neural network	0.813	1.037	0.983	0.837	1.058
PCA smoothing splines	0.898	0.924	1.050	0.989	1.020
	(0.180)	(0.173)	(0.161)	(0.152)	(0.167)

Table 2: Result table for the dataset derived from the foot data. The average cross-validated MSE is presented for each model, together with the average cross-validated tuning parameter value, if tuning of a parameter has been implemented.

In accordance to the results presented in Table 2, the best model obtained for Hypothesis 1 is linear regression together with PCA, which is also the best result overall for the dataset derived from the foot data. The MSE obtained for linear regression with PCA is statistically better than the other results obtained for Hypothesis 1. What should also be noted is that it is also significantly better than when the same model is trained and applied on the other Hypothesis sets. For the other hypotheses, the best results obtained are more similar to each other. These four also had other results closer in range to the best one, than the difference for Hypothesis 1. Smoothing splines and ridge regression, which gave the best results for Hypothesis 2 and Hypothesis 4 respectively, both gave quite similar MSE results for all five hypotheses, which indicates that these two are less affected by the different feature sets than for example linear regression with PCA and MARS cubic, which varied more greatly depending on feature subset. One model that was very affected by the different feature sets was linear regression, with the best value given by Hypothesis 4 (MSE = 0.689) and the worst value given by Hypothesis 1 (MSE = 2.781). When comparing all values, the results were generally better for the last three hypotheses, compared to Hypothesis 1 and 2. The models that performed the worst for the foot dataset for all five hypotheses was linear regression (Hypothesis 1), local regression (Hypothesis 2), and neural network (Hypothesis 3, 4 and 5). Neural network performed poorly for all five hypotheses, whilst local regression performed better for Hypotheses 3, 4 and 5.

4.1.1 Hypothesis 1: All available features

As stated the best model obtained for Hypothesis 1 was linear regression together with PCA, where the nested cross-validation found that the average optimum tuning parameter (number of principal components used) was 15.111. When applying forward selection, the two most significant features was proved to be the dominant frequency from the accelerometer data in the y-direction, and the dominant frequency power from the accelerometer data in the x-direction. These two features were the ones implemented to build the smoothing splines algorithm. The performance of the best model for Hypothesis 1 (linear regression with PCA) can be examined by looking at Figure 5, which displays the average MSE for each of the 18 folds used for the cross-validation. The dotted lines displays the negative and the positive average standard error, while the red line displays the overall average MSE-value, which is the one presented in Table 2. Figure 6 displays the confusion matrix, displaying the added results from all the test sets used by the cross-validation procedure. The confusion matrix has the predicted UPDRS by the model displayed on the horizontal axis, and the true UPDRS on the vertical axis. It should be interpreted such that the diagonal going from the upper left corner to the lower right corner presents the correct classifications, e.g. when the predicted UPDRS matches the true UPDRS. The red boxes indicate incorrect classifications. The grey and blue outline displays the correct (black text) and incorrect classifications (red text) of the different classes (grey boxes) and the summed up result (blue box). This means that for this Hypothesis, the linear regression model correctly classified 59.4 percent of the available test observations.

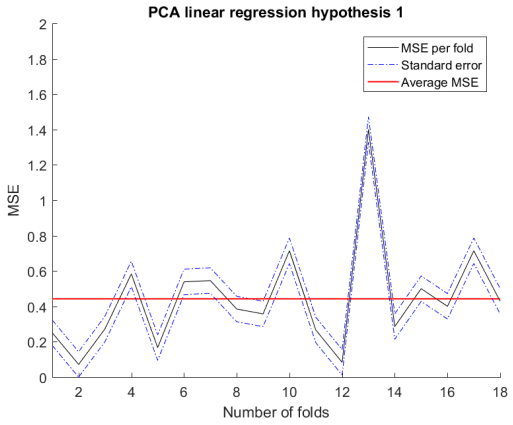


Figure 5: Plot displaying the calculated MSE value for all the 18 folds used for cross-validation. The dotted line presents the standard error related to the overall average MSE value, whilst the overall average MSE is indicated by the red line.

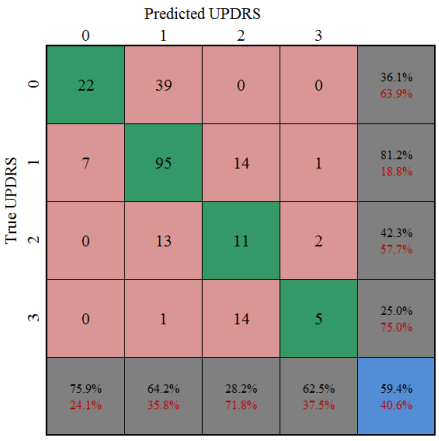


Figure 6: A confusion matrix based on the results from the best obtained model for Hypothesis 1 of the foot data. The blue box displays the overall correct classification percentage (black text) and the incorrect classification percentage (red text).

The displayed curve in Figure 5 indicates that an unusually large MSE value was produced while classifying fold number 13. Without this fold, the average MSE value would be smaller, yielding a better overall result. By examining Figure 6, one may see that the main errors in classification was that true UPDRS 0 was incorrectly predicted as UPDRS 1 in 63.9 percent of the available signals, and that the true UPDRS 2 was falsely set as UPDRS 2 for 14 of the available 20 signals. The best classification was done for UPDRS 1, where 81.2 percent of the signals were correctly classified.

Figure 7 displays two example folds; one showing an example of good accuracy from the model and one showing an example of when the model performed poorly.

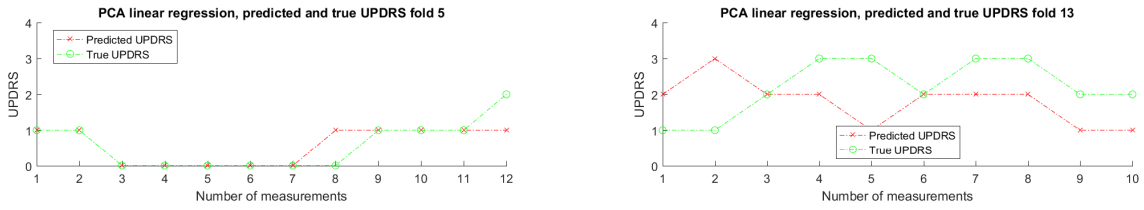


Figure 7: Examples of good and bad classification done for Hypothesis 1. The true and predicted UPDRS is displayed for the number of measurements in a particular fold. Left: fold 5 displaying one of the best classifications. Right: fold 13 displaying one of the worst classifications.

By looking at Figure 7 one may conclude that the performance differed heavily based on the different folds. For the left plot, linear regression with PCA only misclassified two of the observations. For the right plot, which corresponds to the protruding MSE value of fold 13 given in Figure 5, the model instead misclassified all but two observations.

4.1.2 Hypothesis 2: All features from the accelerometer data

As shown in Table 2, the best model for Hypothesis 2 of the foot data, was given by smoothing splines. As for all cases with smoothing splines, the two features used was the two features deemed as the most contributing features from forward selection. These were the dominant frequency and the signal energy, both in the y-direction. The average optimum tuning parameter was chosen to be 0.099. Figure 8 reveals that the calculated MSE differed significantly for the different folds, with maximums in fold 10, 13 and 16.

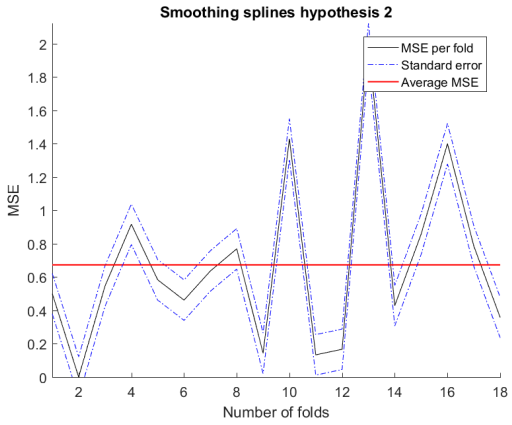


Figure 8: Plot displaying the calculated MSE value for all the 18 folds used for cross-validation. The dotted line presents the standard error related to the overall average MSE value, whilst the overall average MSE is indicated by the red line.

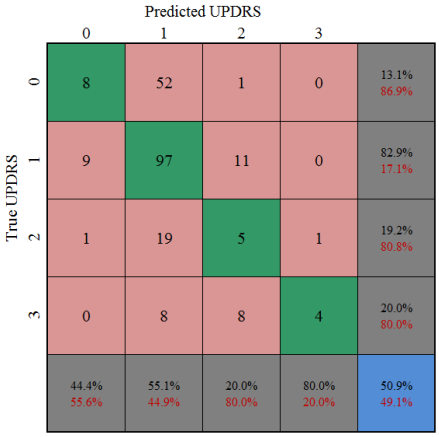


Figure 9: A confusion matrix based on the results from the best obtained model for Hypothesis 2 of the foot data. The blue box displays the overall correct classification percentage (black text) and the incorrect classification percentage (red text).

The confusion matrix presented in Figure 9 gives away that the main difference from the best model for Hypothesis 1 is that more true UPDRS 0 were falsely classified as UPDRS 1. Only 13.1 percent of true UPDRS 1 was correctly classified. This affected the overall performance, which ended up with 50.9 percent of all classification to be correct. Figure 10 displays the classification for fold 9 and fold 10.

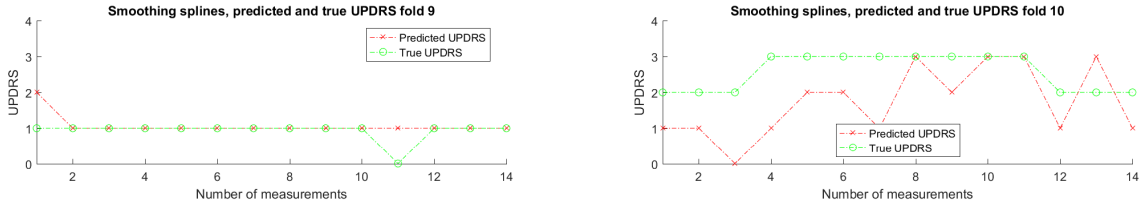


Figure 10: Examples of good and bad classification done for Hypothesis 2. The true and predicted UPDRS is displayed for the number of measurements in a particular fold. Left: fold 9 displaying one of the best classifications. Right: fold 10 displaying one of the worst classifications.

The presented examples in Figure 10 reveal that the classification varies extensively also for this Hypothesis, depending on the different folds.

4.1.3 Hypothesis 3: All features in the y-direction from the accelerometer data

The best model obtained for Hypothesis 3 was MARS cubic, closely followed by smoothing splines. Since a fixed value was set on beforehand for the tuning parameter, no nested cross-validation was used, rather the value 12 was set for the parameter. For the two features used by smoothing splines, forward selection chose the dominant frequency as well as the standard deviation of the dominant frequency power as the two features contributing the most. Figure 11 shows a main peak of the MSE for fold 10. No fold gave perfect classification, indicated by the fact that no MSE value is zero.

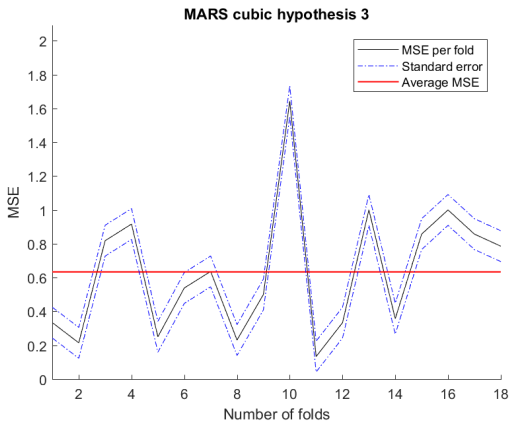


Figure 11: Plot displaying the calculated MSE value for all the 18 folds used for cross-validation. The dotted line presents the standard error related to the overall average MSE value, whilst the overall average MSE is indicated by the red line.

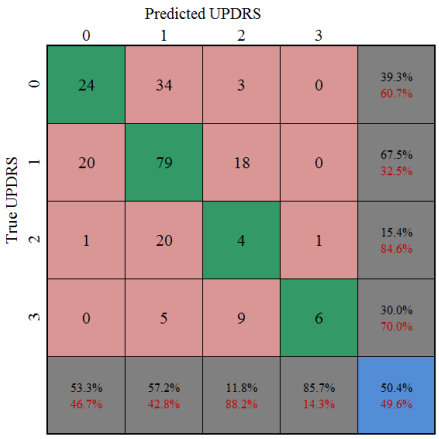


Figure 12: A confusion matrix based on the results from the best obtained model for Hypothesis 3 of the foot data. The blue box displays the overall correct classification percentage (black text) and the incorrect classification percentage (red text).

The confusion matrix in Figure 12 displays a total classification percentage of 50.4 percent correctly classified observations. The largest deviation can be found regarding the classification of UPDRS 3, where 70.0 percent of the available observations were incorrectly classified as UPDRS 1 or UPDRS 2. In Figure 13 the classification for fold 5 and 13 can be seen.

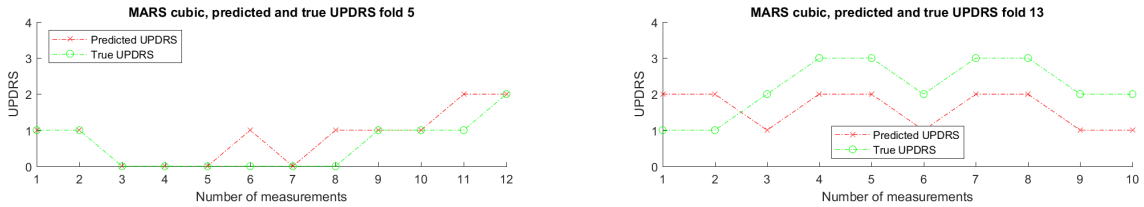


Figure 13: Examples of good and bad classification done for Hypothesis 3. The true and predicted UPDRS is displayed for the number of measurements in a particular fold. Left: fold 5 displaying one of the best classifications. Right: fold 13 displaying one of the worst classifications.

The right example in Figure 13 displays that for fold 13, no correct classification was performed. The fact that fold 13 still however has smaller MSE value than fold 10 (as

shown in Figure 11), is due to unequal sizes of the folds, which is taken into consideration in the equation calculating the MSE. The left example displaying fold 5 shows a better classification, but a few misclassifications are still present.

4.1.4 Hypothesis 4: All features from the gyroscope data

Ridge regression was the model with the best classification performance and lowest average MSE for Hypothesis 4. The corresponding tuning parameter decided by nested cross-validation was calculated as 13.333. The models closest in performance was PCA linear regression, the lasso, smoothing splines, local regression and linear regression, which all had an MSE lower than 0.7. The two best features decided by forward selection, was the standard deviation of the energy content of the second frequency band, applied on the magnitude (XYZ) data, as well as the standard deviation of the energy ratio in the z-direction. As seen in Figure 14, the MSE values were lower for the first eight folds compared to the remaining ones. The worst performance in MSE was reached for folds 10, 13 and 16. The overall classification performance presented in Figure 15 shows that 52.7 percent of all available test observations was appointed the correct UPDRS value.

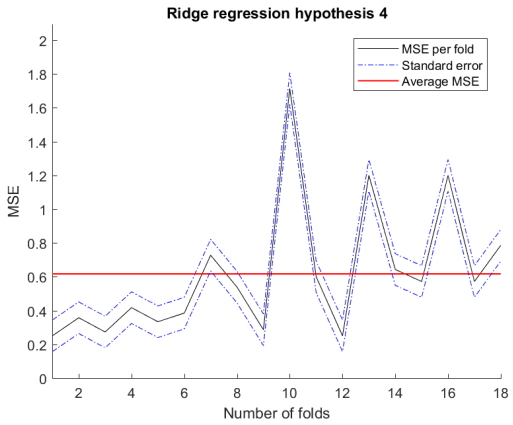


Figure 14: Plot displaying the calculated MSE value for all the 18 folds used for cross-validation. The dotted line presents the standard error related to the overall average MSE value, whilst the overall average MSE is indicated by the red line.

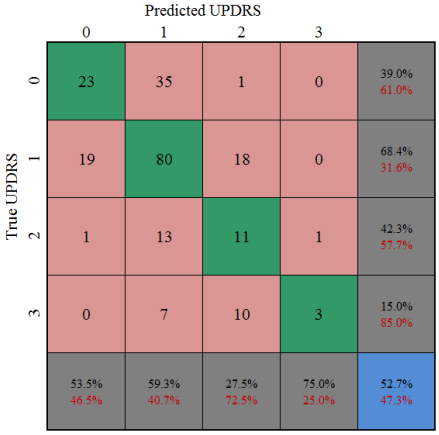
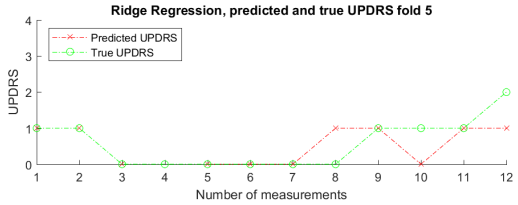


Figure 15: A confusion matrix based on the results from the best obtained model for Hypothesis 4 of the foot data. The blue box displays the overall correct classification percentage (black text) and the incorrect classification percentage (red text).

By further examining the confusion matrix in Figure 15, one may conclude that the best performance was given for the classification of UPDRS 1. For the opposite, the model once again had the largest class error given regarding the classification of UPDRS 3. Figure 16 shows one example of a bad classification and one of a good classification.



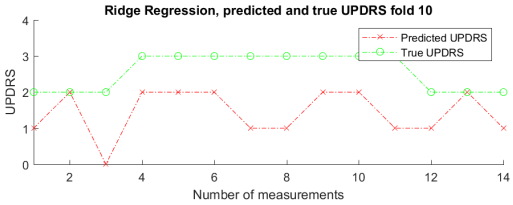


Figure 16: Examples of good and bad classification done for Hypothesis 4. The true and predicted UPDRS is displayed for the number of measurements in a particular fold. Left: fold 5 displaying one of the best classifications. Right: fold 10 displaying one of the worst classifications.

One of the worst classifications is displayed by the right plot in Figure 16, where only two observations were correctly classified. Other than that, the model tended to predict a lower UPDRS value than the true one. The classification worked significantly better for Hypothesis 5, displayed on the left, where three misclassifications were done. However, the misclassifications were still in the same range as the other observations available for that particular fold.

4.1.5 Hypothesis 5: Magnitude features (XYZ) from the accelerometer data

The best model for the last Hypothesis is stated in Table 2 as SVM with a linear kernel, together with the corresponding average tuning parameter of 0.128. For this Hypothesis, a lower amount of models performed almost as good as SVM linear, compared to Hypothesis 4. Two models that also managed to get an average MSE value below 0.7 was linear regression with PCA, and the lasso. The features used for smoothing splines decided by forward selection, was the dominant frequency and the signal energy. Figure 17 shows two deviating peaks for fold 10 and 13. The overall performance is given by the confusion matrix in Figure 18 and reveals that 55.4 percent of all observations were correctly classified.

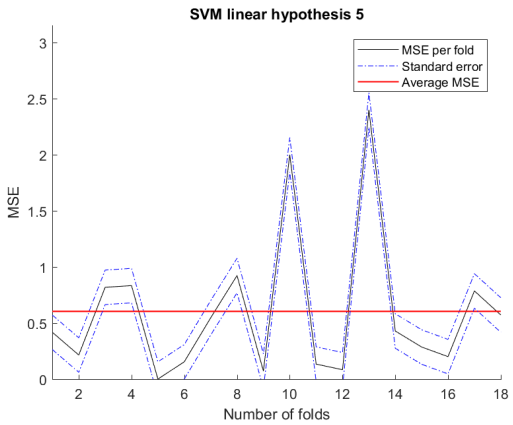


Figure 17: Plot displaying the calculated MSE value for all the 18 folds used for cross-validation. The dotted line presents the standard error related to the overall average MSE value, whilst the overall average MSE is indicated by the red line.

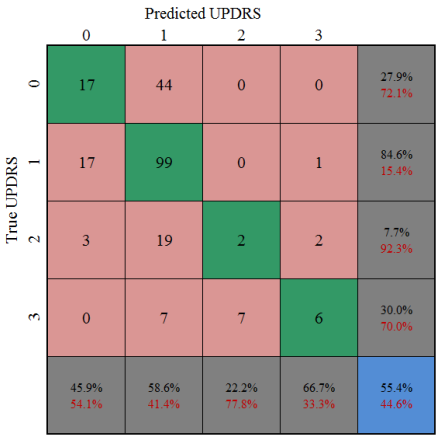


Figure 18: A confusion matrix based on the results from the best obtained model for Hypothesis 5 of the foot data. The blue box displays the overall correct classification percentage (black text) and the incorrect classification percentage (red text).

Figure 18 also reveals that the model SVM linear had the greatest troubles classifying UPDRS 0 and UPDRS 2. However, as seen in Figure 17 the MSE for the different folds varies greatly, which is even further displayed by the examples given in Figure 19:

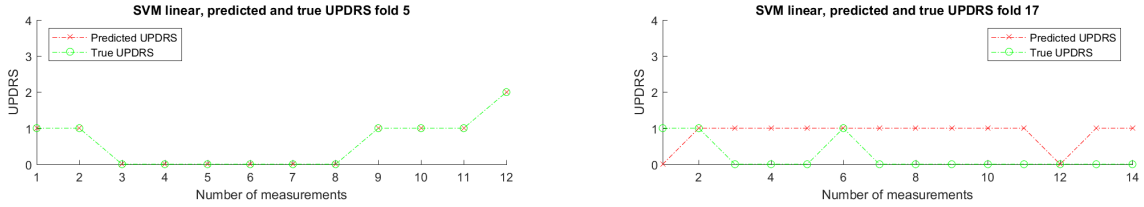


Figure 19: Examples of good and bad classification done for Hypothesis 5. The true and predicted UPDRS is displayed for the number of measurements in a particular fold. Left: fold 5 displaying one of the best classifications. Right: fold 17 displaying one of the worst classifications.

The left plot displays perfect classification, where all the 12 observations were classified correctly by the model. This differs greatly from the right plot, displaying fold 17 where all but three observations were misclassified.

4.2 Foot Data: UPDRS 0 from healthy controls and UPDRS 0 from patients

The second dataset used was a dataset containing measurements from both healthy controls with UPDRS 0 and Parkinson's disease patients with UPDRS 0, e.g. signals deemed as symptom free by the physician. This dataset still stems from the use of ankle sensors during the heel tapping. It should not be possible to distinguish between a healthy controls signal and a patient signal with UPDRS 0, since the latter indicates that the recorded signal is free from bradykinesia. By only comparing healthy control signals with patient signals diagnosed with UPDRS 0, the complexity of the problem is narrowed down from a multi-class problem to a two-class problem. This gives a direct effect on the calculated MSE values, which for this case is much lower. Regarding this case, it should be mentioned that patient observations with UPDRS 0 are denoted as class 1, to distinguish between the two classes during the classification. Observations from healthy controls are therefore denoted as class 0. The numerical results for all models and all hypotheses are presented below in Table 3.

Method	Hyp 1	Hyp 2	Hyp 3	Hyp 4	Hyp 5
Forward selection	0.188	0.285	0.199	0.198	0.186
	(2.056)	(39.889)	(3.056)	(2.000)	(2.944)
Decision tree	0.197	0.170	0.132	0.186	0.132
	(0.016)	(0.016)	(0.009)	(0.031)	(0.015)
K-nearest neighbor	0.066	0.077	0.142	0.138	0.167
	(1.778)	(2.556)	(2.000)	(1.944)	(11.778)
Linear regression	0.465	0.151	0.166	0.241	0.194
Local regression	0.669	0.208	0.190	0.857	0.178
	(0.622)	(0.581)	(0.578)	(0.572)	(0.442)
Smoothing splines FS	0.256	0.157	0.164	0.185	0.173
	(0.143)	(0.060)	(0.138)	(0.074)	(0.070)
MARS linear	0.118	0.129	0.154	0.113	0.177
MARS cubic	0.126	0.147	0.142	0.144	0.125
Neural networks	0.133	0.194	0.221	0.191	0.269
Ridge regression	0.097	0.160	0.155	0.354	0.166
	(35.556)	(5.000)	(0.000)	(12.778)	(12.778)
The lasso	0.084	0.168	0.202	0.139	0.175
	(0.010)	(0.004)	(0.002)	(0.002)	(0.007)
SVM cubic	0.073	0.099	0.215	0.128	0.147
	(0.010)	(0.010)	(0.166)	(0.011)	(0.119)
SVM poly	0.062	0.107	0.141	0.099	0.166
	(0.010)	(0.010)	(0.030)	(0.010)	(0.020)
SVM linear	0.076	0.125	0.188	0.126	0.178
	(0.091)	(0.167)	(0.119)	(0.151)	(0.138)
PCA decision tree	0.294	0.318	0.259	0.349	0.261
	(0.019)	(0.020)	(0.012)	(0.015)	(0.016)
PCA K-nearest neighbor	0.255	0.247	0.242	0.282	0.267
	(18.722)	(19.167)	(9.889)	(25.611)	(17.000)
PCA linear regression	0.114	0.207	0.215	0.240	0.313
	(66.944)	(32.278)	(6.278)	(11.222)	(15.222)
PCA local regression	0.266	0.242	0.157	0.192	0.272
	(0.781)	(0.539)	(0.756)	(0.406)	(0.656)
PCA neural network	0.293	0.264	0.266	0.265	0.284
PCA smoothing splines	0.317	0.287	0.243	0.349	0.259
	(0.178)	(0.179)	(0.160)	(0.171)	(0.171)

Table 3: Result table for the dataset derived from the foot data regarding classification between healthy controls and patients with UPDRS 0. The average cross-validated MSE is presented for each model, together with the average cross-validated tuning parameter value, if tuning of a parameter has been implemented.

The absolute best MSE value for this case, was given by SVM with a polynomial kernel, when applied with Hypothesis 1. This resulted in an overall average cross-validated MSE value of 0.062, which corresponds to 93.9 percent of all test observations being correctly classified. SVM poly also displayed the best result for both Hypothesis 3 and 4 as well, whilst k-nearest neighbor gave the best result for Hypothesis 2. The best result obtained for Hypothesis 5 was given by MARS with a cubic kernel. The worst performance was more widespread for this case than for foot data UPDRS 0 - 3. The worst models in ascending order was PCA neural network, PCA linear regression, PCA decision tree, linear regression and lastly local regression. Generally, models that used PCA performed quite badly for this dataset, indicating that too much information is lost in this case while using two principal components. Some models that performed quite similar for all the five hypotheses, was MARS linear, MARS cubic and PCA k-nearest neighbor, indicating that the performance of these were not so much affected by the different subsets of features.

4.2.1 Hypothesis 1: All available features

As shown in Table 3, the best method for the first Hypothesis is SVM with a polynomial kernel, closely followed by k-nearest neighbor. The overall average cross-validated MSE value obtained by SVM poly was 0.062, with an average tuning parameter of 0.010. SVM with the other two types of kernels, cubic and linear, also performed well, with both alternatives giving an MSE value below 0.08. The worst performance was given by local regression, with a MSE value of 0.669. In the same manner as earlier trials, forward selection was used to choose the two most promising features which were then implemented in building the smoothing splines model. For this Hypothesis, these two features were the dominant frequency in the y-direction from the accelerometer data, and the dominant frequency power in the x-direction, also from the accelerometer data.

Figure 20 displays the calculated MSE value for each fold. From this, it is shown that the worst performance was given for fold 14 and 16. Other than that, several MSE values were kept below 0.1. As stated earlier, SVM with a polynomial kernel gave a very promising overall result, which is presented more in the confusion matrix displayed in Figure 21.

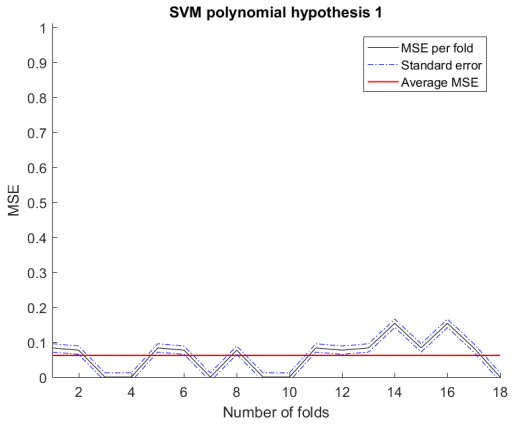


Figure 20: Plot displaying the calculated MSE value for all the 18 folds used for cross-validation. The dotted line presents the standard error related to the overall average MSE value, whilst the overall average MSE is indicated by the red line.

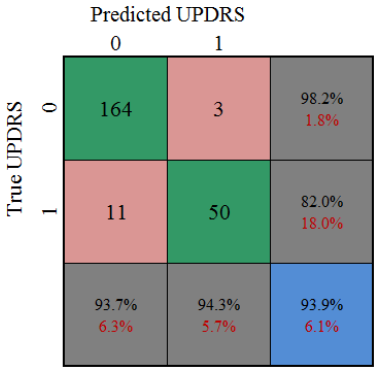


Figure 21: A confusion matrix based on the results from the best obtained model for Hypothesis 1 of the foot data. The blue box displays the overall correct classification percentage (black text) and the incorrect classification percentage (red text).

The confusion matrix states that 93.9 percent of the test observations were correctly classified for all the folds. The model had a harder time classifying class 1 than class 0, but the overall result was quite good for both classes. Again, two examples of classification are shown below in Figure 22, where the performance is shown for fold 10 and fold 14.

Figure 22: Examples of good and bad classification done for Hypothesis 1. The true and predicted class is displayed for the number of measurements in a particular fold. Left: fold 10 displaying one of the best classifications. Right: fold 14 displaying one of the worst classifications.

The left example shows a fold where perfect classification was achieved. The classification shown in the right example, is also very good since it only misclassifies two observations, even though it shows one of the worst examples for this particular model.

4.2.2 Hypothesis 2: All features from the accelerometer data

As displayed in Table 3, the best model obtained for Hypothesis 2 was k-nearest neighbor, which also gave a promising result for Hypothesis 1. The MSE obtained was 0.077 with an average tuning parameter of 2.556. For this Hypothesis, forward selection deemed that the two best features was the energy ratio of the magnitude (XYZ) data, and the standard deviation of the energy content of the first frequency band in the z-direction. The best model KNN had few folds with higher MSE, as displayed in Figure 23, but the overall average MSE was kept low due to perfect classification for several other folds. The confusion matrix for the KNN model in this case is given in Figure 24.

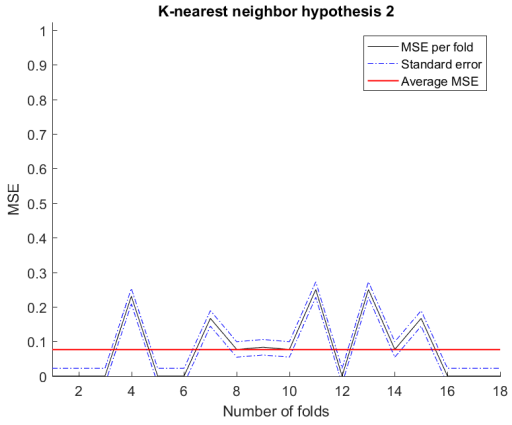


Figure 23: Plot displaying the calculated MSE value for all the 18 folds used for cross-validation. The dotted line presents the standard error related to the overall average MSE value, whilst the overall average MSE is indicated by the red line.

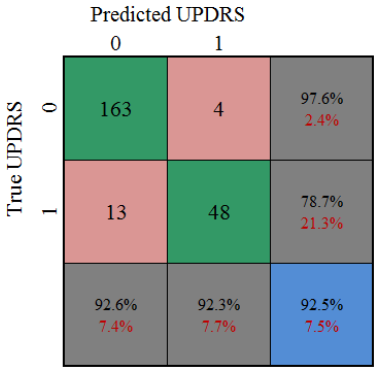


Figure 24: A confusion matrix based on the results from the best obtained model for Hypothesis 2 of the foot data. The blue box displays the overall correct classification percentage (black text) and the incorrect classification percentage (red text).

This confusion matrix shows that 92.5 percent of all the test observations were correctly classified. The partial results shows that four observations with true class 0, and 13 observations with true class 1 was misclassified, as opposed to 163 versus 48 observations being correctly classified. Two examples of classification are displayed in Figure 25.

Figure 25: Examples of good and bad classification done for Hypothesis 2. The true and predicted class is displayed for the number of measurements in a particular fold. Left: fold 2 displaying one of the best classifications. Right: fold 11 displaying one of the worst classifications.

The left plot showing fold 2 displays a case of perfect classification, whilst the right one displaying fold 11 is one of the worst cases. Again, the worst case is not too bad, since only three observations were misclassified.

4.2.3 Hypothesis 3: All features in the y-direction from the accelerometer data

As opposed to the earlier two hypotheses, no model applied on the third Hypothesis managed to get an MSE below 0.1. The best performing method was however SVM poly again, which resulted in an MSE of 0.141 with a tuning parameter value of 0.030. MARS cubic and k-nearest neighbor performed very similar, since both achieved an average MSE of 0.142. In this case, the features from forward selection used in smoothing splines, was the standard deviation of the entropy, and the dominant frequency. The MSE curve presented in Figure 26, had a peak with the highest MSE for fold 17, which affected the average MSE. In Figure 27 the confusion matrix is presented.

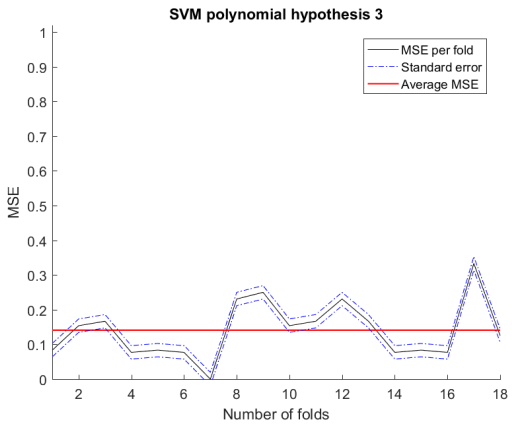


Figure 26: Plot displaying the calculated MSE value for all the 18 folds used for cross-validation. The dotted line presents the standard error related to the overall average MSE value, whilst the overall average MSE is indicated by the red line.

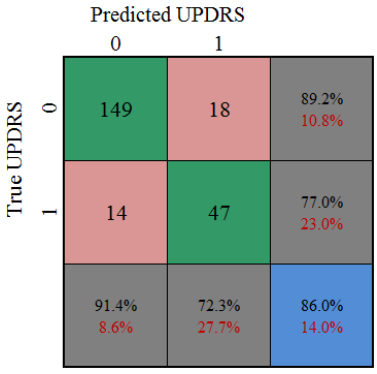


Figure 27: A confusion matrix based on the results from the best obtained model for Hypothesis 3 of the foot data. The blue box displays the overall correct classification percentage (black text) and the incorrect classification percentage (red text).

By looking at the confusion matrix shown in Figure 27 one may conclude that 86.0 percent of all the test observations were correctly classified. Even though this overall is a worse performance than for both Hypothesis 1 and 2, perfect classification was achieved for fold 7, which is displayed in the left plot of Figure 28.

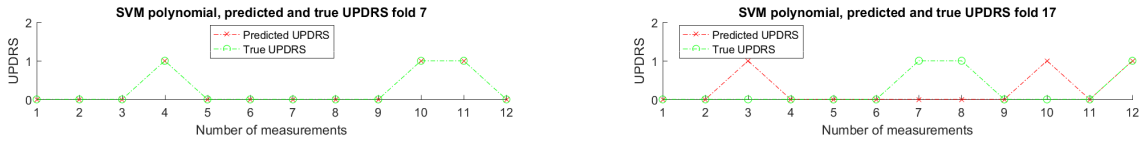


Figure 28: Examples of good and bad classification done for Hypothesis 3. The true and predicted class is displayed for the number of measurements in a particular fold. Left: fold 7 displaying one of the best classifications. Right: fold 17 displaying one of the worst classifications.

The right plot displaying fold 17 however shows a bit worse result, with four test observations being mistakenly classified as the other class.

4.2.4 Hypothesis 4: All features from the gyroscope data

Again, the most promising model was SVM with a polynomial kernel, which resulted in an average cross-validated MSE value of 0.099, with a tuning parameter of 0.010. This result is most closely followed by MARS linear, which resulted in an MSE of 0.113. The features deemed as the two best ones from forward selection, were in this case the standard deviation of the entropy based on the magnitude (XYZ) data, and the entropy in the x-direction. The MSE curve displayed in Figure 29 is quite varied, but perfect classification was achieved for fold 2, 8, 9, 13, and 16. The MSE peaked with the worst results for fold 6, 14 and 15. The confusion plot produced based on SVM poly for Hypothesis 4, is shown in Figure 30.

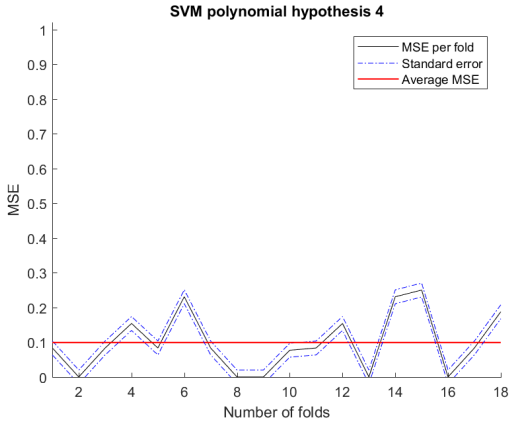


Figure 29: Plot displaying the calculated MSE value for all the 18 folds used for cross-validation. The dotted line presents the standard error related to the overall average MSE value, whilst the overall average MSE is indicated by the red line.

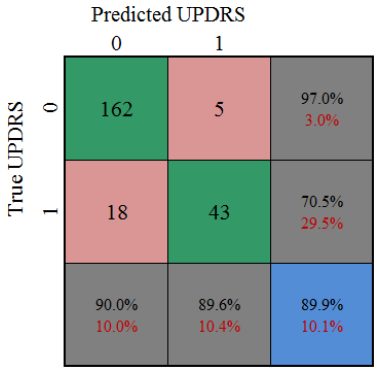


Figure 30: A confusion matrix based on the results from the best obtained model for Hypothesis 4 of the foot data. The blue box displays the overall correct classification percentage (black text) and the incorrect classification percentage (red text).

In Figure 30 it is presented that 89.9 percent of the test observations were correctly assigned to the right class. The classification of observations with true UPDRS 0 gave the best result, with only 3.0 percent of misclassifications. The case that mostly decreased the performance was that 29.5 percent of the test observations with true class 1 was classified as class 0. Figure 31 displays two classifications done for Hypothesis 4.

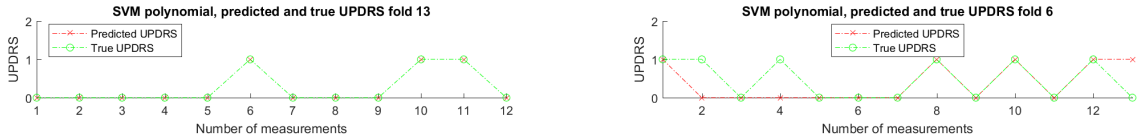


Figure 31: Examples of good and bad classification done for Hypothesis 4. The true and predicted class is displayed for the number of measurements in a particular fold. Left: fold 13 displaying one of the best classifications. Right: fold 6 displaying one of the worst classifications.

The plots given in Figure 31 demonstrate that perfect classification was achieved for fold 13, whilst one of the worst cases shown in fold 6 misclassified 3 out of the 13 observations in that particular fold.

4.2.5 Hypothesis 5: Magnitude features (XYZ) from the accelerometer data

For the last Hypothesis, the most promising model was MARS cubic, with an average cross-validated MSE of 0.125. Since a fixed value of 12 was used as the tuning parameter as described in Section 3.6.6, no average tuning parameter was calculated by nested cross-validation. This particular model performed quite a lot better than the other models for Hypothesis 5, except for decision tree which came quite close with an average MSE of 0.132. For this Hypothesis, forward selection chose the standard deviation of the entropy and the energy ratio as the two most promising features to be used when building the smoothing splines model.

The MSE plot displayed in Figure 32 is kept within quite the same range, except for fold 18 where a peak in the MSE is shown. Perfect classification is achieved for fold 6, 8 and 12. Figure 33 displays the confusion matrix for Hypothesis 5.

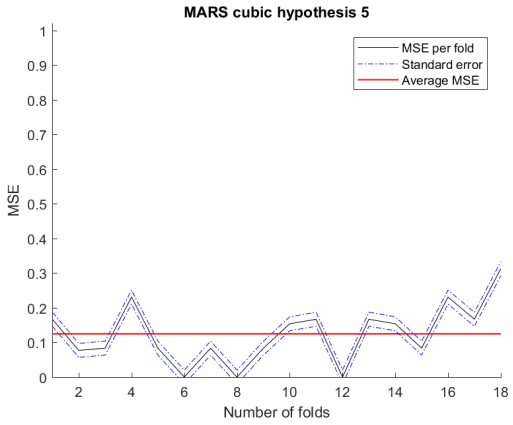


Figure 32: Plot displaying the calculated MSE value for all the 18 folds used for cross-validation. The dotted line presents the standard error related to the overall average MSE value, whilst the overall average MSE is indicated by the red line.

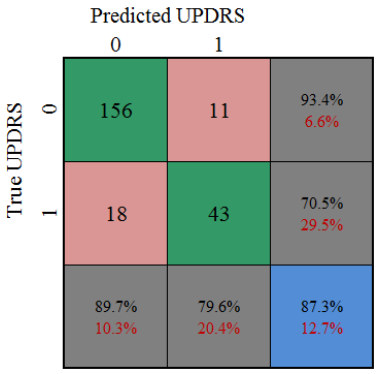


Figure 33: A confusion matrix based on the results from the best obtained model for Hypothesis 5 of the foot data. The blue box displays the overall correct classification percentage (black text) and the incorrect classification percentage (red text).

The confusion matrix in Figure 33 reveals that 87.3 percent of the test observations were assigned to the correct class. The worst misclassification group was observations with true class 0, where 18 out of 61 observations where faulty classified. Figure 34 displays two examples of classification, where fold 8 and 18 are presented.

Figure 34: Examples of good and bad classification done for Hypothesis 5. The true and predicted class is displayed for the number of measurements in a particular fold. Left: fold 8 displaying one of the best classifications. Right: fold 18 displaying one of the worst classifications.

Fold 8 shows, as stated earlier, perfect classification, whilst 5 observations are being wrongly classified for fold 18. This indicates that the number of misclassified observations varied more for MARS cubic applied on Hypothesis 5, compared to the best models for the other hypotheses.

4.3 Hand data: patients with UPDRS 0 - UPDRS 3

The third set of data contains data collected from the sensors attached to the patients' wrists while conducting a rotating movement, switching between the pronation and supination position. Due to this rotating movement, the expected result should be that the gyroscope features plays a major role in the classification, rather than the accelerometer features. The numerical results for all the models are presented within Table 4.

Method	Hyp 1	Hyp 2	Hyp 3	Hyp 4	Hyp 5
Forward selection	1.251	1.430	1.455	1.392	1.098
	(3.056)	(8.278)	(2.500)	(16.332)	(4.770)
Decision tree	1.146	1.096	0.946	0.903	1.207
	(0.039)	(0.047)	(0.048)	(0.036)	(0.046)
K-nearest neighbor	0.758	0.734	0.607	0.768	0.726
	(44.778)	(46.444)	(42.111)	(43.500)	(43.222)
Linear regression	6.102	0.938	0.653	1.701	0.927
Local regression	4.292	1.095	0.751	2.424	0.969
	(0.492)	(0.589)	(0.564)	(0.744)	(0.594)
Smoothing splines FS	0.905	0.642	1.037	1.952	0.606
	(0.103)	(0.027)	(0.077)	(0.023)	(0.064)
MARS linear	1.277	0.930	0.890	0.811	0.833
MARS cubic	1.046	0.901	0.731	1.002	0.786
Neural networks	1.243	1.836	1.402	1.580	1.497
Ridge regression	0.924	0.810	0.640	1.027	0.742
	(392.778)	(447.778)	(150.556)	(595.000)	(523.333)
The lasso	0.729	0.852	0.632	0.887	0.692
	(0.030)	(0.036)	(0.032)	(0.066)	(0.055)
SVM cubic	0.823	0.917	0.829	0.929	0.854
	(0.010)	(0.010)	(0.164)	(0.010)	(0.128)
SVM poly	0.926	0.938	1.207	0.886	0.973
	(0.010)	(0.010)	(0.036)	(0.010)	(0.017)
SVM linear	0.953	0.847	0.686	0.882	0.874
	(0.142)	(0.132)	(0.076)	(0.145)	(0.128)
PCA decision tree	1.018	0.881	0.960	1.113	1.074
	(0.035)	(0.035)	(0.044)	(0.044)	(0.043)
PCA K-nearest neighbor	1.063	0.843	0.719	0.971	0.728
	(17.611)	(31.667)	(42.111)	(34.778)	(279.722)
PCA linear regression	0.803	0.730	1.070	1.340	1.065
	(16.722)	(5.944)	(10.333)	(5.000)	(5.667)
PCA local regression	0.959	0.934	0.924	0.863	1.061
	(0.481)	(0.903)	(0.714)	(0.522)	(0.656)
PCA neural network	1.209	2.134	1.692	2.255	1.239
PCA smoothing splines	0.804	0.667	0.664	0.816	0.730
	(0.158)	(0.149)	(0.146)	(0.143)	(0.135)

Table 4: Result table for the dataset derived from the hand data regarding classification of UPDRS 0 to 3. The average cross-validated MSE is presented for each model, together with the average cross-validated tuning parameter value, if tuning of a parameter has been implemented.

The overall most promising result for the hand dataset was given by the smoothing splines model with an average cross-validated MSE of 0.606 for Hypothesis 5. This was closely followed by k-nearest neighbor which resulted in an MSE of 0.607 for Hypothesis 3. The best results for the three other hypotheses are larger, and given in ascending order by smoothing splines for Hypothesis 2, the lasso for Hypothesis 1, and k-nearest neighbor for Hypothesis 4. What is noticeable is, that despite the predicted result that the gyroscope measurements should play a big role in the classification, out of the top five results, this was the worst. Generally looking at the overall numbers, the results are worse for the hand dataset compared to the foot dataset. This is especially displayed when comparing the range of the top five results from the foot classification, which was 0.442 to 0.673. The same range for the hand classification was 0.607 to 0.768. In the same manner as for the foot data, linear regression performs very poorly for Hypothesis 1, using all the available features. The same manner was displayed for local regression, which also improved its result with the other hypotheses. The five worst Hypothesis were, in ascending order, neural network, neural network using PCA, local regression, and linear regression.

4.3.1 Hypothesis 1: All available features

For the first Hypothesis where all the available features were used, the best obtained result was given by the lasso, which resulted in an average cross-validated MSE of 0.729, and an average tuning parameter of 0.030. The second best model was k-nearest neighbor with an MSE of 0.758. This case contains the largest difference between the best and the worst MSE, where the worst one was given by linear regression with an MSE of 6.102. The features used for smoothing splines were in this case the energy content in the third frequency band, as well as the energy content in the first frequency band, both in the y-direction from the gyroscope data.

Figure 35 displays the MSE for every fold of the best model, and it is easy to spot that the curve contains some really large peaks. The largest one for fold 18 is very deviating compared to the others, which affects the overall MSE result negatively. This result is reflected in the confusion matrix shown in Figure 36.

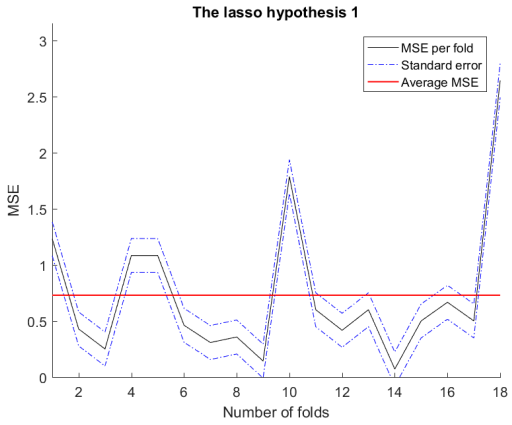


Figure 35: Plot displaying the calculated MSE value for all the 18 folds used for cross-validation. The dotted line presents the standard error related to the overall average MSE value, whilst the overall average MSE is indicated by the red line.

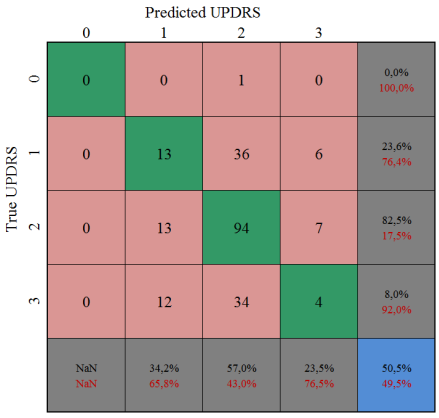


Figure 36: A confusion matrix based on the results from the best obtained model for Hypothesis 1 of the hand data. The blue box displays the overall correct classification percentage (black text) and the incorrect classification percentage (red text).

In the confusion matrix the classification performance is displayed, stating that 50.5 percent of the test observations were correctly classified. The worst misclassification was performed in the case of observations with true UPDRS 3,, where only 8.0 percent of all observations was correctly classified. It may seem a lot that true UPDRS 0 was 100 percent misclassified, but this is a bit misguiding since only one observation with true UPDRS 0 was available.

In the same manner as earlier cases, two examples of good and bad classification for the best model obtained is displayed, here in Figure 37.

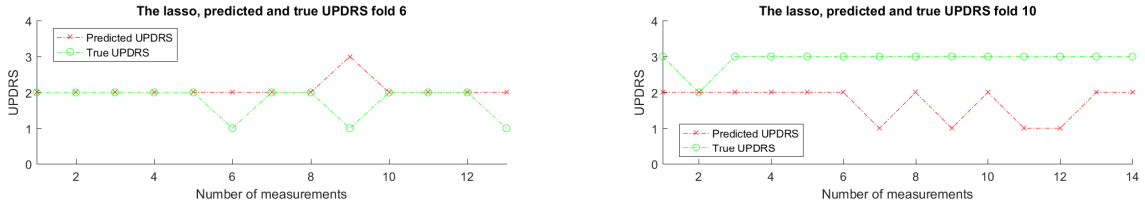


Figure 37: Examples of good and bad classification done for Hypothesis 1. The true and predicted UPDRS is displayed for the number of measurements in a particular fold. Left: fold 6 displaying one of the best classifications. Right: fold 10 displaying one of the worst classifications.

The left plot presents fold number 6, with three misclassifications. The plot to the right which displays a bad case, shows fold 10 where only one of the given observations are correctly classified. For the rest of the observations, the lasso underestimates the class with one or two steps.

4.3.2 Hypothesis 2: All features from the accelerometer data

When the second Hypothesis was used, a noticeable change in performance for the better can be seen while inspecting Table 4. The best result was given by smoothing splines, which resulted in an average cross-validated MSE of 0.642, as well as an average tuning parameter of 0.027. None of the other models performed equally well except for smoothing splines used with PCA, which resulted in an MSE of 0.667. The worst classification result was given by neural network using PCA, which gave the highest MSE value by far. The best features in this case used by smoothing splines, which were again chosen by forward selection, was the standard deviation of the energy content in the third frequency band of the x-direction, and the energy content in the first frequency band, also in the x-direction.

The MSE curve is presented in Figure 38 and the confusion matrix in Figure 39:

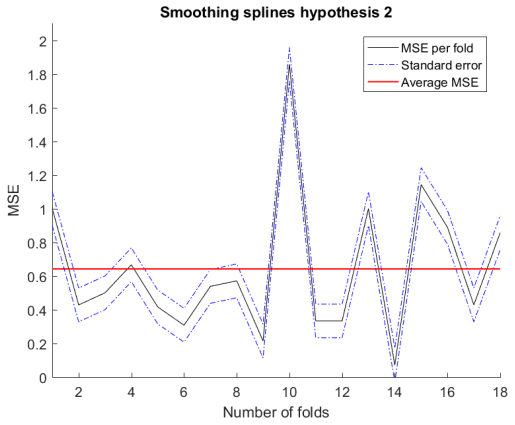


Figure 38: Plot displaying the calculated MSE value for all the 18 folds used for cross-validation. The dotted line presents the standard error related to the overall average MSE value, whilst the overall average MSE is indicated by the red line.

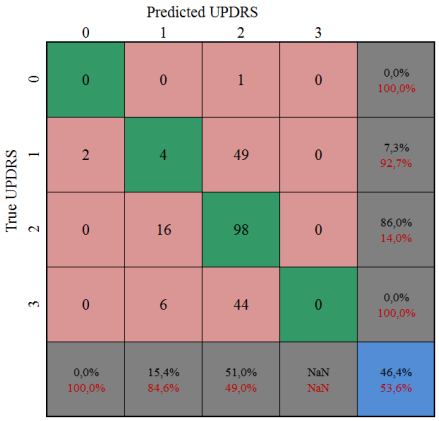


Figure 39: A confusion matrix based on the results from the best obtained model for Hypothesis 2 of the hand data. The blue box displays the overall correct classification percentage (black text) and the incorrect classification percentage (red text).

Here, also a main peak may be distinguished, for fold 10 which greatly influences the average MSE value. Other than that the fold with the lowest MSE value is fold 14. The confusion matrix in Figure 39 displays that 46.4 of the test observations were correctly classified. The model tended to incorrectly predict UPDRS 2 quite often, with a large amount of both true UPDRS 1 and 3 being assigned to the wrong class. The plots in Figure 40 display the difference in classification performance.

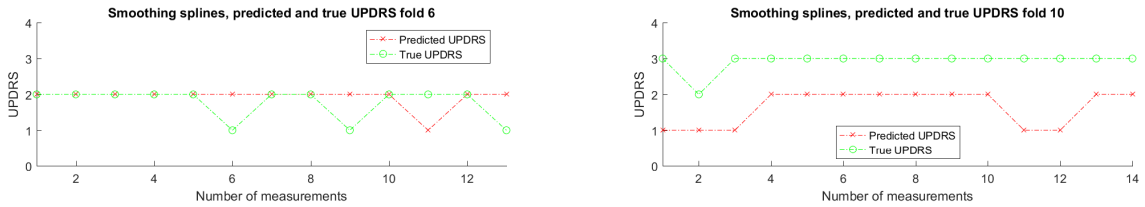


Figure 40: Examples of good and bad classification done for Hypothesis 2. The true and predicted UPDRS is displayed for the number of measurements in a particular fold. Left: fold 6 displaying one of the best classifications. Right: fold 10 displaying one of the worst classifications.

The right plot shows the classification for the worst fold; fold number 10. Here, the model fails to classify any of the observations correctly, constantly underestimating the class with one or two steps. In the left plot the model has been more accurate, with four misclassifications.

4.3.3 Hypothesis 3: All features in the y-direction from the accelerometer data

The most promising model for Hypothesis 3 was given by k-nearest neighbor. It resulted in an average cross-validated MSE of 0.607 with an average tuning parameter of 42.111. For this Hypothesis, there were more models with MSE values close to the best model, than for the earlier two hypotheses. These were linear regression, ridge regression, the lasso,

SVM linear and PCA smoothing splines. For smoothing splines, the two features chosen by forward selection were the standard deviation of the signal entropy, and the dominant frequency.

The MSE plot in Figure 41 shows large variations i the MSE values for the different folds, but no extreme outlier different in range than all the others. The outcome of this case was that 50.0 percent of all test observations was correctly classified, as displayed by the confusion matrix in Figure 42.

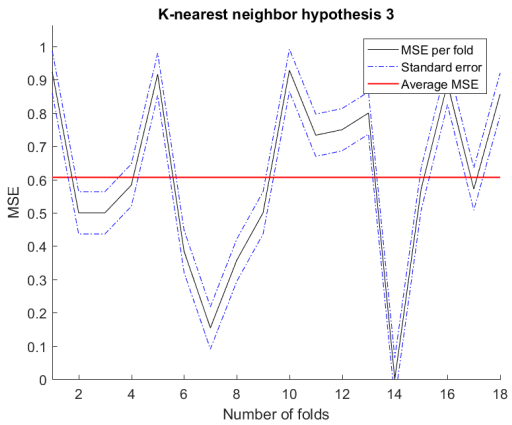


Figure 41: Plot displaying the calculated MSE value for all the 18 folds used for cross-validation. The dotted line presents the standard error related to the overall average MSE value, whilst the overall average MSE is indicated by the red line.

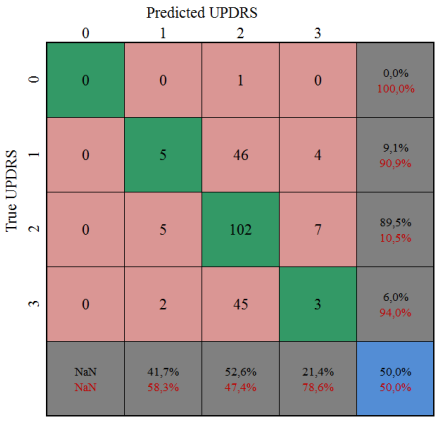


Figure 42: A confusion matrix based on the results from the best obtained model for Hypothesis 3 of the hand data. The blue box displays the overall correct classification percentage (black text) and the incorrect classification percentage (red text).

Again, the greatest misclassification was done by the model assigning observations with true UPDRS 3 and 1, to UPDRS 2. However, observations with true UPDRS 2 was correctly classified for 89.5 percent of the signals. Figure 43 shows one of the best examples and one of the worst examples of classification.

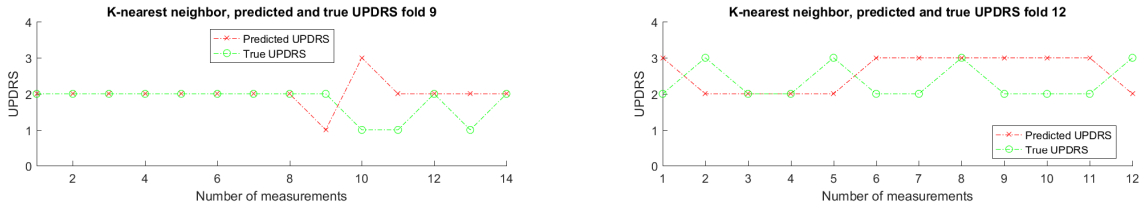


Figure 43: Examples of good and bad classification done for Hypothesis 3. The true and predicted UPDRS is displayed for the number of measurements in a particular fold. Left: fold 9 displaying one of the best classifications. Right: fold 12 displaying one of the worst classifications.

For the left example, fold 9, four out of the 14 observations was given the wrong class, while only three observations was assigned the right class for the right example.

4.3.4 Hypothesis 4: All features from the gyroscope data

For Hypothesis 4, the best average cross-validated MSE value was 0.768, belonging to the k-nearest neighbor model. The corresponding average tuning parameter was set to 43.5. Overall, the performance of the models using Hypothesis 4 was quite off, with the highest MSE value given by local regression. The two features deemed as the most contributing by forward selection, were the energy content in the first and the third frequency band, for the y-direction.

The MSE curve in Figure 44 gets a distinct peak for fold 18, and also quite high MSE for fold 10 and 11. Other than that, the MSE for almost all folds are below the average MSE. The overall result for this model is quite poor, shown by the correct classification rate given by the confusion matrix in Figure 45. This shows that the overall performance was that 45.0 percent of the observations were correctly classified.

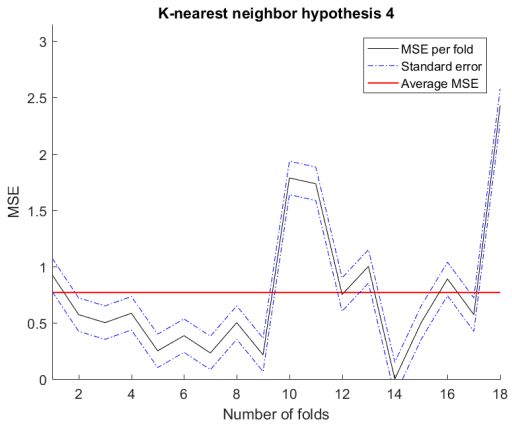


Figure 44: Plot displaying the calculated MSE value for all the 18 folds used for cross-validation. The dotted line presents the standard error related to the overall average MSE value, whilst the overall average MSE is indicated by the red line.

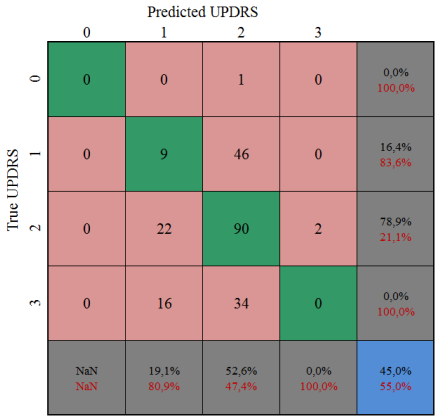


Figure 45: A confusion matrix based on the results from the best obtained model for Hypothesis 4 of the hand data. The blue box displays the overall correct classification percentage (black text) and the incorrect classification percentage (red text).

The main difference from the preceding Hypothesis in performance, is that less observations with true UPDRS 2 was correctly classified. Also, all observations with true UPDRS 3 was misclassified, to UPDRS 1 or 2. Figure 46 presents fold 5 and fold 11 for k-nearest neighbor.

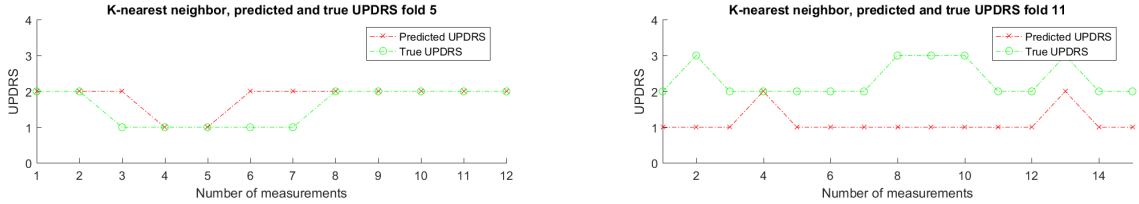


Figure 46: Examples of good and bad classification done for Hypothesis 4. The true and predicted UPDRS is displayed for the number of measurements in a particular fold. Left: fold 5 displaying one of the best classifications. Right: fold 11 displaying one of the worst classifications.

For fold 11 in the right plot, only one of the observations was assigned to the right class. The result for fold 5 is significantly better, with three misclassifications out of 12 observations.

4.3.5 Hypothesis 5: Magnitude features (XYZ) from the accelerometer data

The most promising model for Hypothesis 5, which was smoothing splines, was also the model with the overall lowest average cross-validated MSE value for the hand dataset. This MSE value was 0.606, and the average tuning parameter was 0.064. The model closest in performance was the lasso, with an MSE of 0.692. Here, the two seemingly best features from forward selection were the energy content in the third frequency band, as well as the signal energy. The worst model was neural network, the resulting average MSE from that model was however better than the worst models for the other four hypotheses.

Similar to a few of the other hypotheses, smoothing splines for Hypothesis 5 also gets a peak in the MSE curve for fold 18, as shown by Figure 47. The confusion matrix in Figure 48 gives that 52.7 percent of the classifications was matched correctly to the true value. The result for observations with true UPDRS 1, however was very poor, since none of the 55 observations was correctly classified.

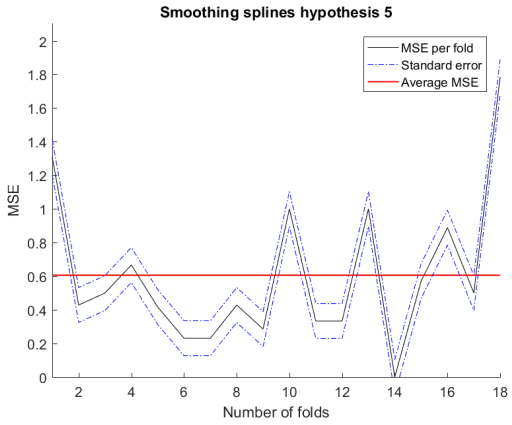


Figure 47: Plot displaying the calculated MSE value for all the 18 folds used for cross-validation. The dotted line presents the standard error related to the overall average MSE value, whilst the overall average MSE is indicated by the red line.

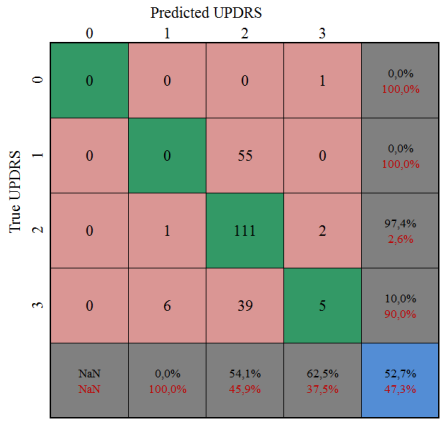


Figure 48: A confusion matrix based on the results from the best obtained model for Hypothesis 5 of the hand data. The blue box displays the overall correct classification percentage (black text) and the incorrect classification percentage (red text).

The plots displayed in Figure 49 shows that one of the best folds correctly classified 10 out of 13 signals, whilst for the worst fold, fold 18, the model only managed to correctly classify one observation.

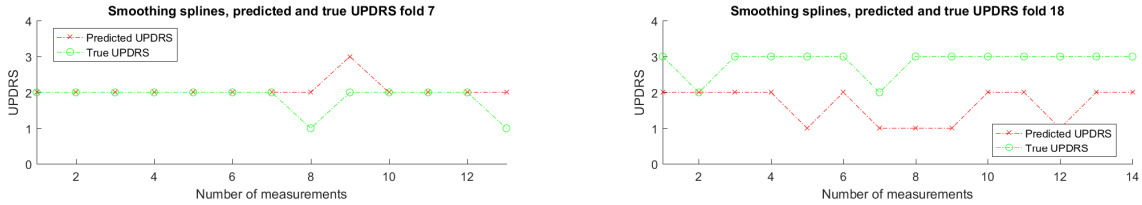


Figure 49: Examples of good and bad classification done for Hypothesis 5. The true and predicted UPDRS is displayed for the number of measurements in a particular fold. Left: fold 7 displaying one of the best classifications. Right: fold 18 displaying one of the worst classifications.

4.4 Foot and hand data: patients with UPDRS 0 - UPDRS 3

The last dataset that was used combined data from both the hand and the foot sensors with UPDRS scores that varied between 0 and 3, meaning that features derived from signals obtained from both of the motor activities are used. The numerical results in the form of the average cross-validated MSE value, together with the average tuning parameter where applicable, is presented within Table 5.

Method	**Hyp 1**	**Hyp 2**	**Hyp 3**	**Hyp 4**	**Hyp 5**
Forward selection	1.064	0.936	1.106	1.207	0.980
	(0.100)	(31.158)	(1.526)	(24.683)	(9.268)
Decision tree	0.638	0.770	0.717	0.839	0.832
	(0.015)	(0.015)	(0.010)	(0.012)	(0.017)
K-nearest neighbor	0.524	0.531	0.665	0.602	0.784
	(3.053)	(5.744)	(7.790)	(2.895)	(4.842)
Linear regression	0.721	0.676	0.725	1.024	0.778
Local regression	0.781	0.743	0.642	0.748	0.745
	(0.616)	(0.474)	(0.408)	(0.408)	(0.426)
Smoothing splines FS	1.088	1.104	1.020	0.976	1.112
	(0.0016)	(0.030)	(0.013)	(0.136)	(0.132)
MARS linear	0.502	0.547	0.537	0.608	0.624
MARS cubic	0.470	0.578	0.580	0.615	0.601
Neural networks	1.925	1.710	1.985	1.911	1.771
Ridge regression	0.494	0.608	0.692	0.625	0.790
	(47.368)	(163.684)	(97.895)	(76.842)	(20.526)
The lasso	0.504	0.616	0.666	0.618	0.780
	(0.016)	(0.016)	(0.051)	(0.010)	(0.001)
SVM cubic	0.432	0.502	0.596	0.546	0.587
	(0.010)	(0.015)	(0.026)	(0.016)	(0.046)
SVM poly	0.477	0.614	0.587	0.520	0.661
	(0.010)	(0.010)	(0.013)	(0.010)	(0.018)
SVM linear	0.608	0.692	0.743	0.668	0.797
	(0.055)	(0.070)	(0.128)	(0.117)	(0.086)
PCA decision tree	1.265	1.359	1.365	1.367	1.143
	(0.023)	(0.030)	(0.020)	(0.015)	(0.013)
PCA K-nearest neighbor	1.031	1.083	1.093	1.074	1.027
	(13.368)	(22.947)	(9.368)	(36.632)	(44.368)
PCA linear regression	0.669	0.708	0.742	0.704	0.800
	(82.790)	(66.737)	(19.543)	(73.526)	(15.790)
PCA local regression	0.771	0.832	0.930	0.794	0.957
	(0.400)	(0.416)	(0.682)	(0.416)	(0.540)
PCA neural network	2.441	2.822	2.172	2.977	2.319
PCA smoothing splines	0.721	0.801	0.848	0.748	0.834
	(0.003)	(0.007)	(0.004)	(0.033)	(0.013)

Table 5: Result table for the dataset derived from the hand data and foot data regarding classification of UPDRS 0 to 3. The average cross-validated MSE is presented for each model, together with the average cross-validated tuning parameter value, if tuning of a parameter has been implemented.

The first noticeable figures when examining these results, are that the worst model for each Hypothesis was neural network using PCA, which gave almost the same average MSE for all five hypotheses. The best result overall for the foot and hand dataset was given by SVM with a cubic kernel, with an average MSE of 0.432. This result was also the absolute best average MSE value achieved for all the three cases where UPDRS 0 to 3 were classified. One should also notice that SVM cubic gave the best average MSE for three out of the five hypotheses. The results obtained for the foot and hand data combined was overall better than for the datasets with only foot or hand data. Even linear regression which has generally performed quite bad for Hypothesis 1 in the earlier cases, gave an average MSE more comparable to the results for the other hypotheses. Other models that gave quite good results compared to the rest was the two different MARS models and SVM polynomial.

4.4.1 Hypothesis 1: All available features

As stated in the previous section, SVM with a cubic kernel was the most promising model, resulting in an average cross-validated MSE of 0.432 and an average tuning parameter of 0.010. The models closest in performance to SVM cubic was SVM poly, ridge regression and MARS cubic. The features that were deemed as the most important ones during forward selection was the energy content in the first and the third frequency band, both for the y-direction from the gyroscope sensor.

By looking at Figure 50, one may see that two peaks are present for fold 5 and 7. However, these two max peaks are significantly smaller than some peaks present for earlier presented cases, which reflects upon the overall performance. The result of SVM cubic gives that 65.1 percent of all the test observations for all folds were correctly assigned to the right class, as presented by the confusion matrix in Figure 51.

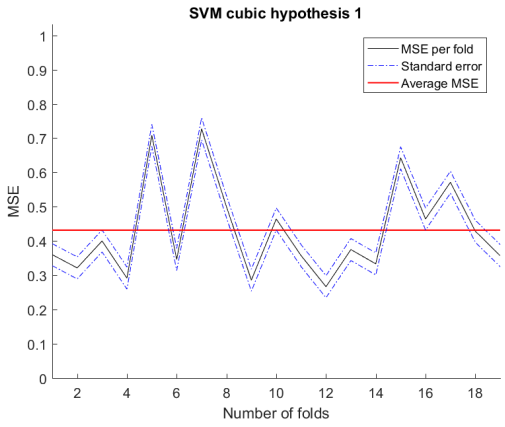


Figure 50: Plot displaying the calculated MSE value for all the 19 folds used for cross-validation. The dotted line presents the standard error related to the overall average MSE value, whilst the overall average MSE is indicated by the red line.

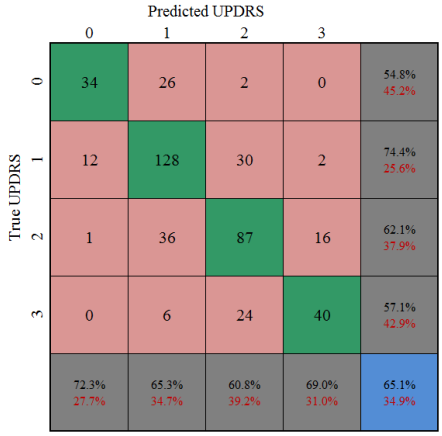


Figure 51: A confusion matrix based on the results from the best obtained model for Hypothesis 1 of the foot and hand data. The blue box displays the overall correct classification percentage (black text) and the incorrect classification percentage (red text).

The confusion matrix also displays that the worst classification case was given for observations with true UPDRS 0, where 45.2 percent were misclassified, mostly assigned wrongly to UPDRS 1. The best classification case was for observations with true UPDRS 1.

Figure 52 displays one of the best and one of the worst classification cases. Now for the foot and the hand data combined, each fold contains a larger amount of observations, due to the overall larger amount of data when merging the two datasets.

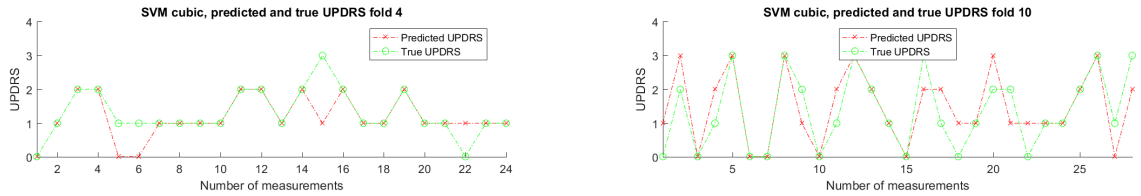


Figure 52: Examples of good and bad classification done for Hypothesis 1. The true and predicted UPDRS is displayed for the number of measurements in a particular fold. Left: fold 4 displaying one of the best classifications. Right: fold 10 displaying one of the worst classifications.

The left plot in Figure 52 shows that for fold 4, four out of 24 test observations were assigned to the wrong class. For one of the worst cases demonstrated by fold 10 (right plot), the number was significantly higher as 12 observations were misclassified.

4.4.2 Hypothesis 2: All features from the accelerometer data

SVM cubic was the best model obtained for Hypothesis 2 as well, with an average cross-validated MSE of 0.502 and an average tuning parameter of 0.015. The models closest in performance was k-nearest neighbor, MARS linear and MARS cubic. For this Hypothesis, the features used for smoothing splines after performing forward selection was the dominant frequency in the y-direction, and the standard deviation of the energy in the z-direction.

The MSE curve in Figure 53 has to maximum peaks for fold 5, and 7. The best MSE derived was about 0.3 for fold 13. The corresponding correct classification percentage was 60.8 percent, as shown by the confusion matrix in Figure 54.

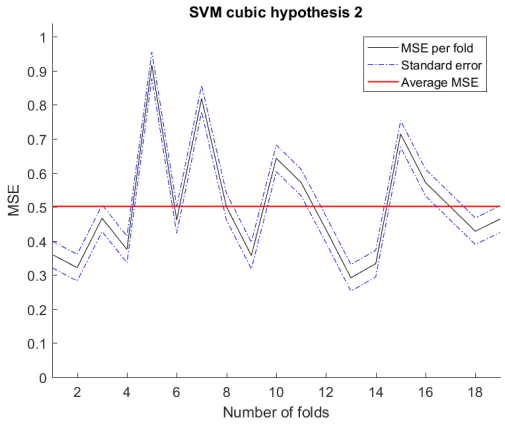


Figure 53: Plot displaying the calculated MSE value for all the 19 folds used for cross-validation. The dotted line presents the standard error related to the overall average MSE value, whilst the overall average MSE is indicated by the red line.

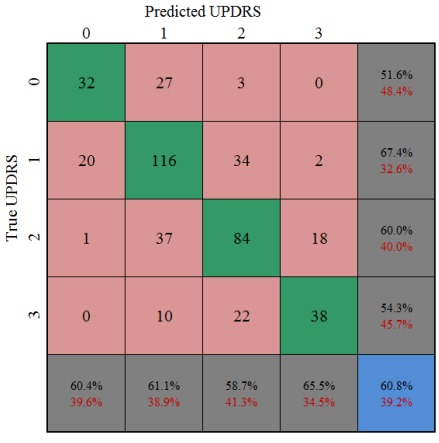


Figure 54: A confusion matrix based on the results from the best obtained model for Hypothesis 2 of the foot and hand data. The blue box displays the overall correct classification percentage (black text) and the incorrect classification percentage (red text).

The confusion matrix also shows that the most accurate classification was achieved for observations with true UPDRS 1. The worst case was for true UPDRS 0, where almost half of the available observations were incorrectly classified. Figure 55 displays a good and a bad classification of Hypothesis 2.

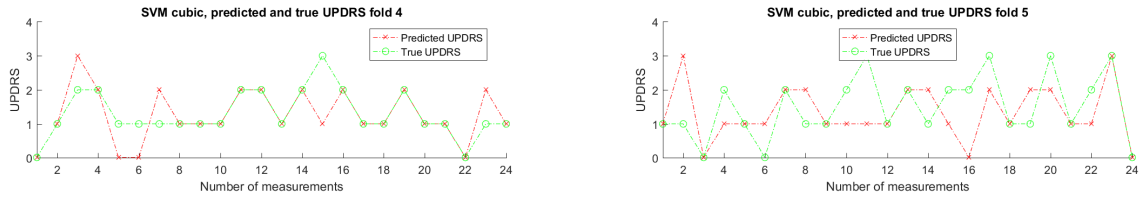


Figure 55: Examples of good and bad classification done for Hypothesis 2. The true and predicted UPDRS is displayed for the number of measurements in a particular fold. Left: fold 4 displaying one of the best classifications. Right: fold 5 displaying one of the worst classifications.

The left plot in Figure 55 shows that out of the available observations in fold 4, 6 observations was assigned to the wrong class. For fold 5, which is the right plot, this number was larger, since 13 observations was misclassified.

4.4.3 Hypothesis 3: All features in the y-direction from the accelerometer data

Table 5 reveals that the best model for the third Hypothesis was MARS linear, with an average cross-validated MSE of 0.537. The second best model was also MARS, but with a cubic kernel instead. As mentioned earlier, the parameter used for MARS was fixed after a first initial trial, to 15. The features deemed as the most promising by forward selection was the dominant frequency and the standard deviation of the dominant frequency power.

The MSE curve in Figure 56 has a clear peak in fold 15, making this the worst MSE value, affecting the average MSE negatively. The resulting MSE values correspond to 58.1 percent of all the test observations being correctly classified, as shown in Figure 57.

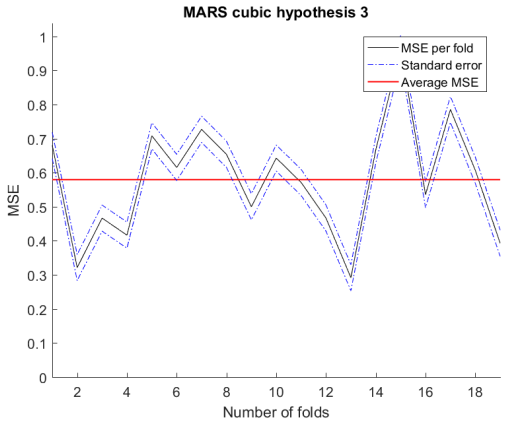


Figure 56: Plot displaying the calculated MSE value for all the 19 folds used for cross-validation. The dotted line presents the standard error related to the overall average MSE value, whilst the overall average MSE is indicated by the red line.

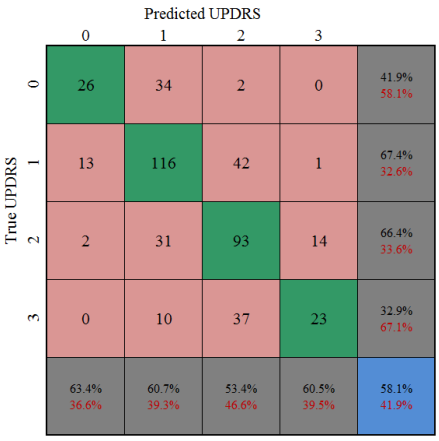


Figure 57: A confusion matrix based on the results from the best obtained model for Hypothesis 3 of the foot and hand data. The blue box displays the overall correct classification percentage (black text) and the incorrect classification percentage (red text).

As for many of the earlier cases, the best classification was achieved for observations with true UPDRS 1. Two classification examples are shown in Figure 58, one for fold 13 and one for fold 10.

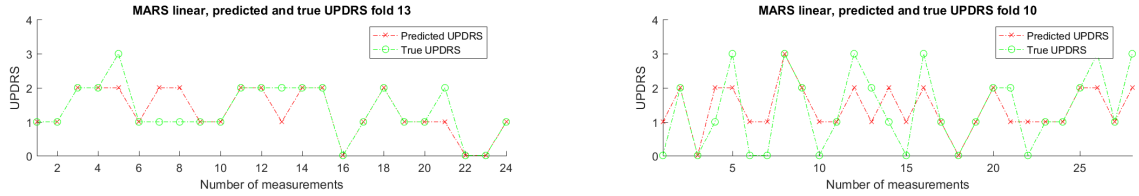


Figure 58: Examples of good and bad classification done for Hypothesis 3. The true and predicted UPDRS is displayed for the number of measurements in a particular fold. Left: fold 13 displaying one of the best classifications. Right: fold 10 displaying one of the worst classifications.

As shown for fold 13 in the left plot, five observations were misclassified out of the total 24 of this particular fold. The case displayed by fold 10 in the right plot, had 15 observations assigned to the wrong class.

4.4.4 Hypothesis 4: All features from the gyroscope data

SVM was again the most promising model, but this time with a polynomial kernel with an average cross-validated MSE of 0.520. The gap to the remaining models where quite large, except for the MARS cubic model. The best features obtained by forward selection and used by smoothing splines, was the dominant frequency energy in the x-direction, and the signal energy in the y-direction.

Figure 59 has one deviating peak for fold 10 where the MSE value reached a value slightly above 1.0. The resulting percentage of successful classifications was 62.4 percent, as displayed in the confusion matrix in Figure 60.

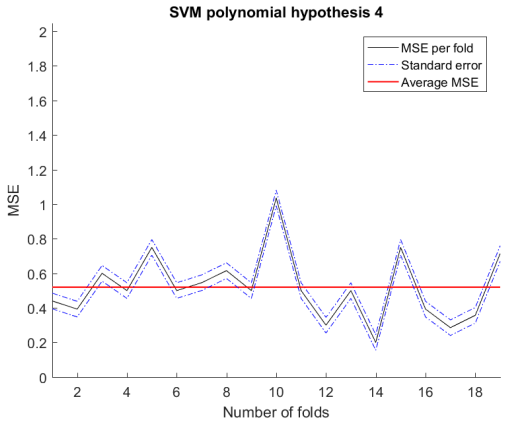


Figure 59: Plot displaying the calculated MSE value for all the 19 folds used for cross-validation. The dotted line presents the standard error related to the overall average MSE value, whilst the overall average MSE is indicated by the red line.

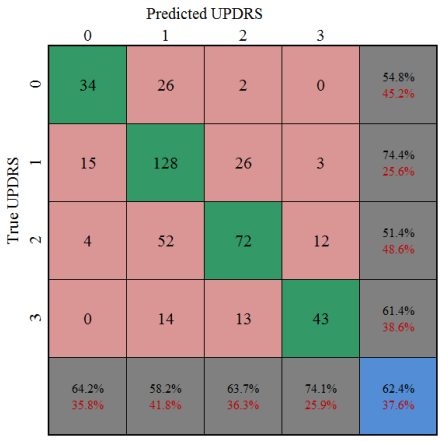


Figure 60: A confusion matrix based on the results from the best obtained model for Hypothesis 4 of the foot and hand data. The blue box displays the overall correct classification percentage (black text) and the incorrect classification percentage (red text).

Again, the model had the largest troubles with classifying observations with true UPDRS 1, followed by the classification of true UPDRS 3. The good and poor examples displayed in Figure 61, shows that for the good example illustrated by fold 12, 24 out of 30 observations where correctly classified. The difference compared to fold 15, which illustrates poor classification for the model, is that for that particular fold, 15 of the observations were misclassified.

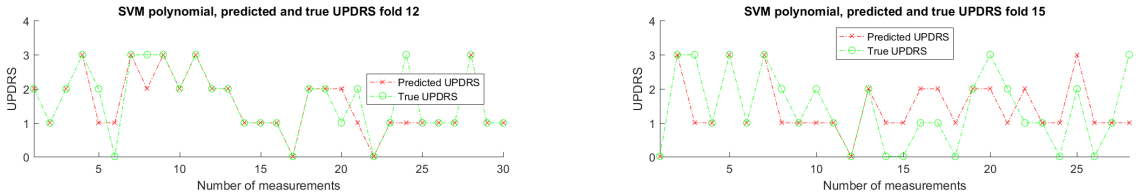


Figure 61: Examples of good and bad classification done for Hypothesis 3. The true and predicted UPDRS is displayed for the number of measurements in a particular fold. Left: fold 12 displaying one of the best classifications. Right: fold 15 displaying one of the worst classifications.

4.4.5 Hypothesis 5: Magnitude features (XYZ) from the accelerometer data

For the last Hypothesis, the best model obtained was once again SVM cubic, with an average cross-validated MSE value of 0.587 and an average tuning parameter value of 0.046. The two best features according to forward selection was in this case the energy content in the third frequency band, as well as the signal energy.

As seen in Figure 62, the worst MSE is signified by the peak present for fold 5. Other than that, most of the MSE values for each fold were kept below 0.8. For this Hypothesis, 59,5 percent of the observations were correctly classified, as seen in Figure 63.

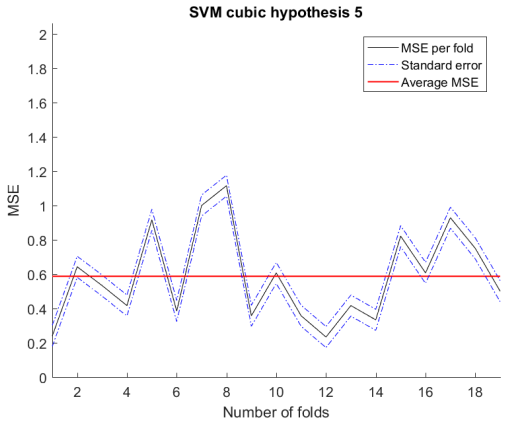


Figure 62: Plot displaying the calculated MSE value for all the 19 folds used for cross-validation. The dotted line presents the standard error related to the overall average MSE value, whilst the overall average MSE is indicated by the red line.

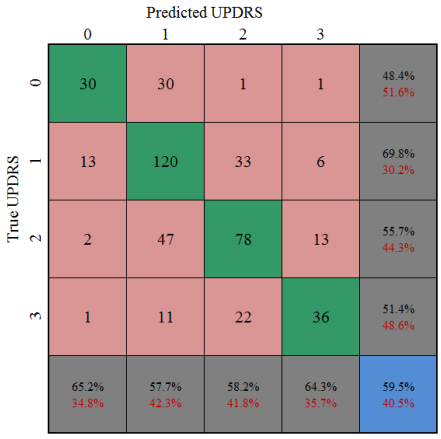


Figure 63: A confusion matrix based on the results from the best obtained model for Hypothesis 5 of the foot and hand data. The blue box displays the overall correct classification percentage (black text) and the incorrect classification percentage (red text).

As shown in the confusion matrix, the worst classification case was regarding the classification of observations with true UPDRS 0, where most of these were assigned to UPDRS 1.

For Figure 64, the left plot shows fold 1 where 6 out of 25 observations were incorrectly classified. For fold 2, which is a worse example, 15 observations were misclassified.

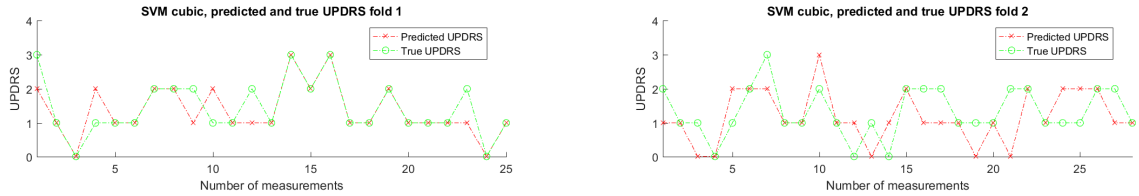


Figure 64: Examples of good and bad classification done for Hypothesis 5. The true and predicted UPDRS is displayed for the number of measurements in a particular fold. Left: fold 1 displaying one of the best classifications. Right: fold 2 displaying one of the worst classifications.

4.5 Summary of results

Table 6 displays a summary of the best results for all the models obtained, as well as for which case and Hypothesis the results are from. The first part of the table displays the best MSE obtained for each model during the classification of Parkinson's disease patients diagnosed with UPDRS 0 - UPDRS 3, which was conducted on foot data, hand data, and the combination of the two. The second part displays the best MSE obtained during the classification between healthy controls and Parkinson's disease patients with UPDRS 0.

Method	MSE	Hypothesis	MSE	Hypothesis
Forward selection	0.773	Foot, hyp 3	0.186	Hyp 5
Decision tree	0.638	Foot+hand, hyp 1	0.132	Hyp 5
K-nearest neighbor	0.524	Foot+hand, hyp 1	0.066	Hyp 1
Linear regression	0.653	Hand, hyp 3	0.151	Hyp 2
Local regression	0.642	Foot+hand, hyp 3	0.178	Hyp 5
Smoothing splines FS	0.606	Hand, hyp 5	0.157	Hyp 2
MARS linear	0.502	Foot+hand, hyp 1	0.113	Hyp 4
MARS cubic	0.470	Foot+hand, hyp 1	0.125	Hyp 5
Neural networks	1.243	Hand hyp 1	0.133	Hyp 1
Ridge regression	0.494	Foot+hand, hyp 1	0.097	Hyp 1
The lasso	0.504	Foot+hand, hyp 1	0.084	Hyp 1
SVM cubic	0.432	Foot+hand, hyp 1	0.073	Hyp 1
SVM poly	0.477	Foot+hand, hyp 1	0.062	Hyp 1
SVM linear	0.604	Foot, hyp 5	0.076	Hyp 1
PCA decision tree	0.867	Foot, hyp 3	0.259	Hyp 3
PCA K-nearest neighbor	0.719	Hand, hyp 3	0.242	Hyp 3
PCA linear regression	0.442	Foot, hyp 1	0.114	Hyp 1
PCA local regression	0.771	Foot+hand, hyp 1	0.157	Hyp 3
PCA neural network	0.813	Foot, hyp 1	0.264	Hyp 2
PCA smoothing splines	0.595	Hand, hyp 3	0.243	Hyp 3

Table 6: Table displaying the best results for each method, first part displays the best average cross-validated MSE for all cases classifying patient data with UPDRS 0 to UPDRS 3. The second part displays the best average cross-validated MSE for the classification of healthy controls versus patients with UPDRS 0, which was conducted only for the foot data.

While looking at the results in Table 6, it indicates that generally the best dataset was the combination of both foot and hand data, for the Hypothesis were all available features where used (Hypothesis 1). For the case of classifying between healthy controls and patients with UPDRS 0, the results were a bit more mixed, however it is clear that Hypothesis 1 was the best for that case as well, with the magnitude data (Hypothesis 5) on second place. The dataset with gyroscope features (Hypothesis 4) only gave the best result for one method, which was MARS linear.

The best possible result for UPDRS 0 to 3 was obtained by the method SVM cubic, which resulted in an average cross-validated MSE of 0.432, which corresponds to 65.1 percent of correctly classified observations. Regarding the classification between healthy controls and UPDRS 0, the best possible MSE was given by SVM with a polynomial kernel. This resulted in an MSE value of 0.062 which corresponds to 93.9 percent of all the observations being correctly classified.

5 Discussion

This work has investigated the accuracy of a set of machine learning algorithms for the quantification and classification of signals obtained from Parkinson's disease patients. The results differed quite a lot, both between different hypotheses and between the selected models. While some models performed better for one setting, the same model might not always have performed well in others. From the results obtained, it was not possible to find any clear distinction between the results obtained from hand data and foot data, but slightly better results were obtained with the dataset combining hand and foot data than the other two. A possible explanation to this could be that training on both hand and foot signals could yield larger understanding regarding the characteristics of the signals. Another probably contributing aspect of this, is that the improved results probably also stems from the fact that the combined dataset contains almost double the amount of observations. This yields that the models have a lot more training data to fit the classifier.

One limitation of this work has been that the amount of data available has been sparse. Even though a lot of signals have been collected, the total number is quite small in machine learning measures. Cross-validation is a nice method to apply whilst working with a limited data set, but it is always better to work with a larger number of observations which might have affected the outcome of this thesis. The small portion of this dataset is reflected in the uneven classes where UPDRS 1 and 2 are dominating whilst the others are quite small. As stated earlier in this report, this resulted in signals diagnosed with UPDRS 4 being discarded. It would of course be preferable to build the classification based on the full range of the UPDRS scale, but this could mean that the classification of UPDRS is based on personal traits present in the signals since the signals with UPDRS 4 stem from the same patients for hand and foot data. Even though observation weights have been applied for methods were this is needed, it would still be preferable to train the different methods on a larger dataset with classes of equal size. Not only would this be good for the accuracy, but also because more data from more patients would mean a smaller risk that the classification is correlated to the specific patient where it stems from.

One particular model that were especially affected by the dimension of the feature set was linear regression, which overall yielded very poor results for Hypothesis 1. This was the overall case for several of the linear models used, even though linear regression stood out. This could be explained due to the fact that a lot of linear model tend to suffer from the curse of dimensionality, meaning their accuracy tend to drop with a larger feature space, especially if the number of features approach the number of observations.

Another comment regarding the linear models is that it was noted with some of these that they were more sensitive to the difference in class sizes in the available data. This was also more obvious regarding the hand data, were some models tended to give very poor results only guessing outputs based on the most significant class available. This performance was improved with the implementation of observation weights during the training phase of the model.

Even with observation weights implemented, it is noticeable that both for the hand and foot data, several models tend to be able to more often classify UPDRS 1 and 2 correctly, than class 0 and 3. Again, this probably depends on the amount of training data available, since UPDRS 1 and 2 are the most frequent classes. A result from this is that the models do not have that many observations belonging to UPDRS 0 and 3 to train on, yielding that they tend to assign to the middle classes on the UPDRS scale, even with observation weights. Especially the class of UPDRS 3 is small compared to the other classes, which

makes it difficult for the models to train properly on different kinds of observations with UPDRS 3.

From the confusion matrices presented, one may see that it is more common that the models classify incorrectly by one step on the UPDRS scale, rather than for example assigning an observation with UPDRS3 to class 0. Maybe with more training data and fine-tuning, the accuracy of assigning to a certain class could be improved. One procedure sometimes used when working with a dataset with large varieties in class sizes, is to trim the most numerous classes to even out the size of them in regards to the smaller classes. However, this approach did not seem like an option for this work, since the amount of observations were already quite sparse.

One interesting aspect when considering the summary of results in Section 4.5 is that for the classification between UPDRS 0 to 3, it is clear that the best case was for the combined dataset with foot and hand data, and Hypothesis 1. But considering the other two datasets with either hand or foot data, other hypotheses, mainly Hypothesis 3 with accelerometer features in the y-direction is the one that yielded the most promising results. This shows that generally better results were obtained for the hypotheses with smaller feature sets, except that this changed when combining the hand and foot data. Then, Hypothesis 1 containing all available features yielded better results.

One model that performed quite bad overall was neural networks, which gave poor results both with and without PCA. This is quite surprising since neural networks is often presented as an extremely versatile algorithm applicable on a large amount of problems, and it has earlier been used in studies using machine learning to classify dyskinesia symptoms. One thought about this could be that no parameter tuning were done in this implementation, rather the built-in matlab functions handled the tuning of all available parameters itself. The only parameter decided by hand was the number of nodes in the hidden layer, which was decided based on one trial with nested cross-validation due to the computational expense in time of using nested cross-validation for all trials. It might have been possible to gain better results if a more hands on approach had been implemented, with the model being more adjusted to this particular problem. The amount of training data available could also be a factor, since a neural network might require a large number of feed-forward and feed-backward iterations to converge to a good model. One last suggestion could also be performing trials with a different set ups of the network, such as a different amount of hidden layer for example.

The most surprising acquired result stems from the classification between healthy controls and diagnosed patients with UPDRS 0. As stated in Section 4.2, theoretically the algorithms should have a hard time with this kind of classification. This is because patients diagnosed with UPDRS 0 by the physician are considered free from bradykinesia, and should therefore in theory resemble signals from a healthy subject. This result however, was very good as the best model obtained for this case correctly classified 93.9 percent of all the test observations. This result indicates that it is possible that machine learning algorithms are able to detect signal characteristics which are related to Parkinson's disease, which the physicians are not able to detect through visual inspection. Of course, since this was just investigated for one dataset it would require more testing before final conclusions may be done, but the results are still promising. One could argue that there is a slight difference between the age range of the two groups, and that this affects the results. The age of the healthy subjects varied between 50 to 76 years old, whilst the age range for Parkinson's disease patients was 61 to 82 years old. Due to this it would be recommended to conduct new trials based with subjects in the same age range, to exclude any affects from the slight shift in age between the two

groups. Another aspect to consider is that the signals from Parkinson's patients might still be affected from other symptoms than bradykinesia, causing the machine learning models to distinguish the signals in other ways.

When discussing the difference in result ranges between the two classification cases (patients with UPDRS 0 - 3 and healthy controls versus UPDRS 0), it should be mentioned that the main difference in range of the MSE value is affected by the complexity of the problem. Between healthy controls and UPDRS 0, the problem is narrowed down to a two-class problem, in contrast to the four-class problem of the classification of patients with UPDRS 0 - 3. Therefore it is not strange that the MSE values are overall a lot smaller for that case since more classes enhance the complexity of a classification problem.

By further discussing particular methods implemented for this work, one remark that should be done is that it would be preferable to use nested cross-validation for each implementation of MARS cubic and MARS linear, as well as for neural networks. As stated in Section 3.6.6, nested cross-validation was only used once to find an optimum tuning parameter, and that value was later used for all runs. For MARS cubic with nested cross-validation the code took almost two weeks to run, which makes it unfeasible to use for all trials in a project of this size. One could argue that this implementation of MARS with a fixed tuning parameter after the first trial is not the optimum one, but it still produced quite good results for most hypotheses.

From the results regarding the classification between healthy controls and UPDRS 0 presented in Section 4.2, it is clear that the models using PCA did overall perform quite weak. One suggestion could be that this case would require a larger number of principal components to be used, due to a loss in information when using the 2 first principal components decided for this work.

When discussing certain results for the folds presented, one may notice that for the foot data, a lot of the models presented tend to get a higher MSE score for fold 10 and 13. In the same manner, fold 18 for the hand data seems to yield worse results generally. This could possibly be explained by the fact that the folds in this work were divided based on the patient data, e.g. signals from one patient constitute one fold. Maybe, certain personal characteristics of a particular patient affects the possibility to classify that certain patient, yielding overall worse results for some of the folds.

The overall results from this study are mixed. As stated, the result from the classification between healthy controls and UPDRS 0 was promising yielding a high success rate for the classification. The results from the other classification cases between UPDRS 0 and 3 gave less accurate results, with the best result being a percentage of 65.1 percent of all test observations being correctly classified. A study conducted by Mera et. al., came to the conclusion that their best result for the classification of dyskinesia was an MSE value of 0.3 [47], which may be compared to our best MSE for bradykinesia, which was 0.432. It should however be noted regarding the result of UPDRS 0 to 3, that several models experienced a high peak in MSE value for one or a couple of folds, which affects the average MSE negatively. Application on another dataset, or further investigation regarding the difficulties with these particular folds could improve the accuracy and overall performance of the models used for this work.

6 Conclusion

The purpose of this work was to investigate the possibility of using machine learning for the classification of the Parkinson's disease symptom bradykinesia. The outcome of this study will contribute to a project working with the development of a frequent assessment system from continuous movements in the patients daily life. All data was collected from both gyroscope and accelerometer sensors during two specific body movements, to further investigate if one particular sensor type signal or movement gives more promising results.

This work has resulted in some promising results regarding the ability to diagnose physical symptoms of Parkinson's disease by using machine learning algorithms. The more surprising part of this is the results regarding the classification between healthy controls and patients with UPDRS 0, since UPDRS 0 should mean that the patient is free from bradykinesia. The conclusion regarding this is however that the machine learning algorithms were indeed able to distinguish certain traits characteristic for the signals belonging to Parkinson's disease patients. One suggestion for applications where this could be of interest is when diagnosing early stage possible Parkinson's disease patients. Then machine learning could be an aid in diagnosing Parkinson's even though the patient have not yet encountered any bradykinesia symptoms, or possibly other motor symptoms.

The results for the classification between UPDRS 0 to 3 were less accurate, but we believe that it is still good enough to motivate further development and investigation regarding improvements of the obtained models. The recommendation considering the results obtained, would for further studies be to select the most promising models obtained in this work, and further develop them with better fine-tuning, and adapting them on fresh datasets.

As stated in the discussion a conclusion regarding the datasets is that it seems like the dataset containing both data from the sensors that collected data from the hand and data from the sensors that collected data from the foot are giving better results than the datasets where there are only foot data or hand data. The Hypothesis that generates the best result is Hypothesis 1. This is surprising since Hypothesis 1 consists of the most features, which in many application makes certain methods perform worse than for a smaller feature set. An explanation to this could be that all features together might complement each other for some models in such a way so that the result gets very good. The next best Hypothesis is Hypothesis 3 which contains accelerometer features in the y-direction. Hypothesis 5, containing magnitude features from the accelerometer data, is the third best Hypothesis. These two hypotheses, Hypothesis 3 and 5, are the two sets of features containing a low number of features and therefore they are, as expected among the best hypotheses since they lower the chance of models suffering from the curse of dimensionality.

By examining the presented results regarding what features that forward selection deemed as the most contributing ones, some features are picked more often. These are features regarding the signal energy, entropy, energy contents in the frequency bands (especially band 1 and 3), and the dominant frequency, and also the standard deviation of these. Therefore it is recommended for future work to include these kind of features when training classifiers for the classification of Parkinson's disease symptoms.

The overall most promising results are given by the support vector machine model, which when implemented with a cubic kernel resulted in an average cross-validated MSE of 0.432 for the combined foot and hand data, when implemented with Hypothesis 1. This corresponds to 65.1 percent of the test observations being correctly classified. For the classification between healthy controls and UPDRS 0, SVM with a polynomial kernel resulted in an average MSE of 0.062, yielding 93.9 percent of all observations to be assigned to the right class.

Considering the discussion, it should also be concluded that more training data is desirable, since the available dataset is quite small. Especially more signal examples from UPDRS 0, 3 and 4 should be acquired to be able to train the models on more even classes.

Since a large variety of machine learning methods has been implemented and tested in this work, there has been less time for fine-tuning and modulation of each individual algorithm. For future work, it would be interesting to further develop the most promising methods found in this study, to see if improvements can be done. Since work has also been done based on signals retrieved during a relatively small time window, our recommendation would be to work further with this project and apply the most promising machine learning algorithms to more continuous data collected from the patients daily life. By supplying a patient with a wristband or other sensor device, this could in the future be used to allow the physicians to get a clearer view on symptom fluctuation during a patients daily life. A more continuous observation could also help with the adjustment of medicine if it could be tracked that the symptoms get more severe during certain times during the day, which could aid the patient in countering the physical symptoms. A more even adjustment of medicine would mean an increase in the patients quality of life, reducing the extreme symptom periods.

References

[1] Acreo. Multimodal motor symptoms quantification platform for individualized Parkinson's disease treatment. https://www.acreo.se/projects/multimodal-motor-symptoms-quantification-platform-for-individualized-parkinsons-disease, 2017. [Online; accessed 2017–02–10].

[2] G.G. Goetz et. al. MDS-UPDRS: Ny version av UPDRS-skalan nu lanserad. http://www.orionpharmaneurologi.se/Archive/Nummer-1---2009/Parkinson/MDS-UPDRS-Ny-version-av-UPDRS-skalan-nu-lanserad/, 2008. [Online; accessed 2017–02–10].

[3] Hjärnfonden. Parkinsons sjukdom. http://www.hjarnfonden.se/om-hjarnan/diagnoser/parkinsons-sjukdom/?gclid=CMfZ7q6ShdCFQyLsgodsKgA-g. [Online; accessed 2017–02–10].

[4] Parkinsonguiden. Orsaker till parkinsons sjukdom. http://parkinsonguiden.parkinsonforbundet.se/vad-ar-parkinson/orsaker-till-parkinsons-sjukdom/, 2014. [Online; accessed 2017–02–17].

[5] Matthew Menza. Combating depression in parkinson's disease. http://www.pdf.org/en/combating_depression, 2009. [Online; accessed 2017–02–23].

[6] Parkinsonförbundet. Behandling och mediciner. http://www.parkinsonforbundet.se/meny2/Om%20Parkinsons%20sjukdom/Behandling%20och%20mediciner.html, 2012. [Online; accessed 2017–02–23].

[7] Unified parkinson's disease rating scale. http://img.medscape.com/fullsize/701/816/58977_UPDRS.pdf, 2006. [Online; accessed 2017–02–17].

[8] P. Bonato et. al. Data mining techniques to detect motor fluctuations in parkinson's disease. *Proceedings of the 26th annual international conference of the IEEE EMBS*, pages 4766–4769, 2004.

[9] J. Cancela et. al. A comprehensive motor symptom monitoring and management system; the bradykinesia case. *Proceedings of the 32nd annual international conference of the IEEE EMBS*, 2:1008–1011, 2010.

[10] S. Patel et. al. Monitoring motor fluctuations in patients with parkinson's disease using wearable sensors. *IEEE transactions on information technology in biomedicine*, 13(6):864–873, 2009.

[11] M. G. Tsipouras et. al. An automated methodology for levodopa-induced dyskinesia; assessment based on gyroscope and accelerometer signals. *Artificial intelligence in medicine*, 55:127–135, 2012.

[12] B. Ratner. *Statistical and machine-learning data mining: techniques for better predictive modeling and analysis of big data.* Taylor & Francis, 2012.

[13] S.L. Salzberg. On comparing classifiers: pitfalls to avoid and a recommended approach. *Data mining and knowledge discovery*, 1:317–328, 1997.

[14] S. Theodoridis & K. Koutroumbas. *Pattern recognition.* Elsevier, 2009.

[15] S. Mallick. Bias-variance tradeoff in machine learning. http://www.learnopencv.com/bias-variance-tradeoff-in-machine-learning/, 2017. [Online; accessed 2017–04–17].

[16] G. James et. al. *An introduction to statistical learning*. Springer, 2013.

[17] Vinnova. A single center, open label, single dose study to assess a multimodal motor symptoms quantification platform (musyq) in relation to plasma levodopa, carbidopa and 3-o-methyldopa concentrations after administration of levodopa/carbidopa in patients with parkinson´s disease who experience motor fluctuations. 2015.

[18] Shimmer3. http://www.shimmersensing.com/products/shimmer3. [Online; accessed 2017–05–04].

[19] Li Tan. *Digital Signal Processing - Fundamentals and Applications*. Elsevier, 2008.

[20] R.Schilling G. Heinzel, A. Rüdiger. pectrum and spectral density estimation by the discrete fouriervtransform (dft), including a comprehensive list of window functions and some new at-top windows. Technical report, Max-Planck-Institut für Gravitationsphysik, (Albert-Einstein-Institut), Teilinstitut Hannover, 02 2002.

[21] R.I. Griffiths et. al. Automated assessment of bradykinesia and dyskinesia in parkinson's disease. *Journal of Parkinson's disease*, 2:47–55, 2012.

[22] M. Wendebourg. Symptom quantification for Parkinson's disease. Master thesis in Systems, Control and Mechatronics, Chalmers University of Technology, 2016.

[23] K.R. Rao et. al. *Fast Fourier Transform: Algorithms and applications*. Springer, 2010.

[24] D.P. Redmond & F.W. Hegge. Observations on the design and specification of a wrist-worn human activity monitoring system. *Behavior research methods, instruments & computers*, 17(6):659–669, 1985.

[25] T. Heida et. al. Power spectral density analysis of physiological, rest and action tremor in parkinson's disease patients treated with deep brain stimulation. *Journal of neuro-engineering and rehabilitation*, 10(1):70–81, 2013.

[26] A. Salarian et. al. Quantification of tremor and bradykinesia in parkinson's disease using a novel ambulatory monitoring system. *IEEE transactions on biomedical engineering*, 54(2):313–322, 2007.

[27] N.L.W. Keijsers et. al. Detection and assessment of the severity of levodopa-induced dyskinesia in patients with parkinson's disease by neural networks. *Movement disorders*, 15(6):1104–1111, 2000.

[28] M. G. Tsipouras et. al. On automated assessment of levodopa-induced dyskinesia in parkinson's disease. *Proceedings of the 33rd annual international conference of the IEEE EMBS*, pages 2679–2682, 2011.

[29] Y. Dodge. *The concise encyclopedia of statistics*. Springer, 2008.

[30] Mathworks. Spectrogram: spectrogram using short-time fourier transform. https://se.mathworks.com/help/signal/ref/spectrogram.html, 2017. [Online; accessed 2017–05–25].

[31] M.M. Goodwin. *Springer handbook of speech processing*. Springer Berlin Heidelberg, 2008.

[32] H. Dai et. al. Quantitative assessment of parkinsonian bradykinesia based on an inertial measurement unit. *Biomedical engineering online*, 14:68–81, 2015.

[33] A.J. Manson et. al. An ambulatory dyskinesia monitor. *Journal of Neurology, Neurosurgery, and Psychiatry*, 68(2):196–201, 2000.

[34] S.H. Roy et. al. High-resolution tracking of motor disorders in parkinson's disease during unconstrained activity. *Movement Disorders*, 28(8):1080–1087, 2013.

[35] J. Cancela et. al. A telehealth system for parkinson's disease remote monitoring. the perform approach. *35th annual international conference of the IEEE EMBS*, pages 7492–7495, 2013.

[36] P.R. Burkhard et. al. Quantification of dyskinesia in parkinson's disease: Validation of a novel instrumental method. *Movement disorders*, 14(5):754–763, 1999.

[37] J. Gour et. al. Movement patterns of peak-dose levodopa-induced dyskinesias in patients with parkinson's disease. *Brain research bulletin*, 74(1):66–74, 2007.

[38] W. Maetzler et. al. Quantitative wearable sensors for objective assessment of parkinson's disease. *Movement Disorders*, 28(12):1628–1637, 2013.

[39] C. Ley et. al. Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median. *Journal of experimental social psychology*, 49(4):764–766, 2013.

[40] I. Wasito & B. Mirkin. Nearest neighbour approach in the least-squares data imputation algorithms. *Information sciences*, 169(1):1–25, 2005.

[41] Weisberg. Variable Selection and Regularization. `http://bigdata.unl.edu/documents/ASA_Workshop_Materials/Variable%20Selection%20and%20Regularization.pdf`, 2012. [Online; accessed 2017–06–29].

[42] Gints Jekabsons. Locally weighted polynomials toolbox for matlab/octave. `http://www.cs.rtu.lv/jekabsons/regression.html`. [Online; accessed 2016–05–12].

[43] A. Araveeporn. The estimation of smoothing parameter using smoothing techniques on nonparametric regression. *Silpakorn university science & technical journal*, 6(1):14–22, 2012.

[44] StatSoft Inc. Electronic statistics textbook. `http://www.statsoft.com/textbook/`, 2013. [Online; accessed 2017–04–26].

[45] Gints Jekabsons. Areslab: Adaptive regression splines toolbox for matlab/octave. `http://www.cs.rtu.lv/jekabsons/regression.html`. [Online; accessed 2016–05–10].

[46] K.L. Priddy & P.E. Keller. *Artificial neural networks: an introduction*. SPIE, 2005.

[47] T.O. Mera et. al. Quantitative assessment of levodopa-induced dyskinesia using automated motion sensing technology. *Proceedings of the 34th annual international conference of the IEEE EMBS*, pages 154–157, 2012.

Appendices

Result plots Foot data UPDRS 0 to UPDRS 3

The results from the first dataset and each hypothesis are below presented in tables.

Hypothesis 1

In table 7 the results from running the first hypothesis on the dataset containing foot data with UPDRS between 0 and 3.

Method	**Cross-validated MSE ± SE**	**Tuning Parameter ± SE**
Forward selection	0.863± 0.162	41.111 ± 6.123
Decision tree	1.377 ± 0.353	0.043 ± 0.002
K-nearest neighbor	0.777 ± 0.176	15.056 ± 1.531
Linear regression	2.781 ± 0.619	-
Local regression	2.632 ± 0.487	0.678 ± 0.046
Smoothing splines FS	0.749 ± 0.131	0.150 ± 0.014
MARS linear	1.194 ± 0.196	15.444 ± 1.183
MARS cubic	1.103 ± 0.138	11.898 ± 2.004
Neural networks	1.703 ± 0.360	44.444 ± 1.661
Ridge regression	0.661 ± 0.106	37.778 ± 9.273
The lasso	0.819 ± 0.239	0.012 ± 0.003
SVM cubic	0.834 ± 0.161	0.010 ± $4.207*10^{-19}$
SVM poly	0.817 ± 0.153	0.010 ± $4.207*10^{-19}$
SVM linear	0.771 ± 0.154	0.128 ± 0.013
PCA decision tree	1.088 ± 0.280	0.096 ± 0.010
PCA K-nearest neighbor	0.962 ± 0.264	38.111 ± 1.747
PCA linear regression	0.442 ± 0.072	15.111 ± 0.387
PCA local regression	0.964 ± 0.246	0.414 ± 0.071
PCA neural network	0.813 ± 0.179	-
PCA smoothing splines	0.898 ± 0.194	0.180 ± 0.011

Table 7: Table displaying the resulting average cross-validated MSE together with the attached average standard error (SE), and the average tuning parameter with the attached average standard error. This table shows results for hypothesis 1 for foot data from Parkinson's disease patients.

Hypothesis 2

In table 8 the results from running the second hypothesis on the dataset containing foot data with UPDRS between 0 and 3.

Method	Cross-validated MSE ± SE	Tuning Parameter ± SE
Forward selection	0.986 ± 0.257	19.222 ± 3.333
Decision tree	1.348 ± 0.260	0.030 ± 0.003
K-nearest neighbor	0.997 ± 0.270	15.722 ± 1.793
Linear regression	2.626 ± 1.628	-
Local regression	3.283 ± 2.022	0.672 ± 0.042
Smoothing splines FS	0.673 ± 0.122	0.099 ± 0.019
MARS linear	1.069 ± 0.169	-
MARS cubic	0.891 ± 0.168	-
Neural networks	1.614 ± 0.370	-
Ridge regression	0.777 ± 0.182	70.556 ± 24.165
The lasso	0.776 ± 0.170	0.012 ± 0.003
SVM cubic	0.842 ± 0.135	$0.011 \pm 5.556{*}10^{-4}$
SVM poly	0.804 ± 0.155	$0.010 \pm 4.207{*}10^{-19}$
SVM linear	0.792 ± 0.170	0.171 ± 0.010
PCA decision tree	0.895 ± 0.266	0.046 ± 0.010
PCA K-nearest neighbor	0.854 ± 0.271	40.389 ± 2.527
PCA linear regression	0.557 ± 0.124	17.111 ± 1.019
PCA local regression	0.949 ± 0.267	0.833 ± 0.040
PCA neural network	1.037 ± 0.272	-
PCA smoothing splines	0.924 ± 0.215	0.173 ± 0.008

Table 8: Table displaying the resulting average cross-validated MSE together with the attached average standard error (SE), and the average tuning parameter with the attached average standard error. This table shows results for hypothesis 2 for foot data from Parkinson's disease patients.

Hypothesis 3

In table 9 the results from running the third hypothesis on the dataset containing foot data with UPDRS between 0 and 3.

Method	Cross-validated MSE ± SE	Tuning Parameter ± SE
Forward selection	0.773± 0.194	8.056 ± 0.802
Decision tree	1.046 ± 0.201	0.022 ± 0.003
K-nearest neighbor	0.873 ± 0.212	10.611 ± 1.079
Linear regression	0.800 ± 0.201	-
Local regression	0.918 ± 0.203	0.670 ± 0.052
Smoothing splines FS	0.673 ± 0.122	0.099 ± 0.019
MARS linear	0.953 ± 0.189	-
MARS cubic	0.634 ± 0.091	-
Neural networks	1.560 ± 0.370	-
Ridge regression	0.706 ± 0.207	26.111 ± 4.289
The lasso	0.755 ± 0.188	0.016 ± 0.003
SVM cubic	0.917 ± 0.174	0.137 ± 0.014
SVM poly	0.991 ± 0.147	0.104 ± 0.013
SVM linear	0.790 ± 0.189	0.159 ± 0.012
PCA decision tree	0.867 ± 0.247	0.101 ± 0.003
PCA K-nearest neighbor	1.052 ± 0.322	39.167 ± 2.368
PCA linear regression	0.750 ± 0.186	8.056 ± 0.698
PCA local regression	0.954 ± 0.261	0.675 ± 0.050
PCA neural network	0.983 ± 0.258	-
PCA smoothing splines	1.049 ± 0.211	0.161 ± 0.012

Table 9: Table displaying the resulting average cross-validated MSE together with the attached average standard error (SE), and the average tuning parameter with the attached average standard error. This table shows results for hypothesis 3 for foot data from Parkinson's disease patients.

Hypothesis 4

In table 10 the results from running the fourth hypothesis on the dataset containing foot data with UPDRS between 0 and 3.

Method	Cross-validated MSE ± SE	Tuning Parameter ± SE
Forward selection	0.791 ± 0.126	32.667 ± 5.107
Decision tree	1.071 ± 0.191	0.031 ± 0.004
K-nearest neighbor	0.973 ± 0.271	40.611 ± 2.470
Linear regression	0.689 ± 0.096	-
Local regression	0.683 ± 0.091	0.700 ± 0.033
Smoothing splines FS	0.631 ± 0.088	0.153 ± 0.012
MARS linear	1.076 ± 0.174	-
MARS cubic	0.868 ± 0.101	-
Neural networks	1.363 ± 0.188	-
Ridge regression	0.617 ± 0.094	13.333 ± 4.918
The lasso	0.648 ± 0.098	0.004 ± 0.001
SVM cubic	1.014 ± 0.230	0.014 ± 0.001
SVM poly	1.066 ± 0.237	$0.010 \pm 4.207{*}10^{-19}$
SVM linear	0.903 ± 0.212	0.157 ± 0.013
PCA decision tree	0.881 ± 0.245	0.046 ± 0.010
PCA K-nearest neighbor	1.032 ± 0.257	44.222 ± 1.680
PCA linear regression	0.697 ± 0.087	65.167 ± 0.853
PCA local regression	0.995 ± 0.241	0.594 ± 0.057
PCA neural network	0.837 ± 0.184	-
PCA smoothing splines	0.989 ± 0.219	0.152 ± 0.016

Table 10: Table displaying the resulting average cross-validated MSE together with the attached average standard error (SE), and the average tuning parameter with the attached average standard error. This table shows results for hypothesis 4 for foot data from Parkinson's disease patients.

Hypothesis 5

In table 11 the results from running the fifth hypothesis on the dataset containing foot data with UPDRS between 0 and 3.

Method	Cross-validated MSE ± SE	Tuning Parameter ± SE
Forward selection	0.810 ± 0.188	9.111 ± 1.275
Decision tree	1.233 ± 0.244	0.021 ± 0.002
K-nearest neighbor	0.820 ± 0.188	8.833 ± 0.968
Linear regression	0.798 ± 0.179	-
Local regression	0.790 ± 0.132	0.650 ± 0.050
Smoothing splines FS	0.707 ± 0.132	0.098 ± 0.012
MARS linear	0.737 ± 0.151	-
MARS cubic	0.739 ± 0.149	-
Neural networks	1.374 ± 0.295	-
Ridge regression	0.740 ± 0.187	18.889 ± 4.269
The lasso	0.697 ± 0.178	0.017 ± 0.007
SVM cubic	0.813 ± 0.137	0.159 ± 0.010
SVM poly	0.889 ± 0.126	0.057 ± 0.013
SVM linear	0.604 ± 0.154	0.128 ± 0.011
PCA decision tree	1.022 ± 0.256	0.063 ± 0.011
PCA K-nearest neighbor	0.922 ± 0.270	29.000 ± 2.249
PCA linear regression	0.668 ± 0.165	8.222 ± 0.888
PCA local regression	0.922 ± 0.272	0.844 ± 0.040
PCA neural network	1.058 ± 0.245	-
PCA smoothing splines	1.020 ± 0.225	0.167 ± 0.012

Table 11: Table displaying the resulting average cross-validated MSE together with the attached average standard error (SE), and the average tuning parameter with the attached average standard error. This table shows results for hypothesis 5 for foot data from Parkinson's disease patients.

Result plots foot data healthy controls versus UPDRS 0

The results from the second dataset and each hypothesis are below presented in tables.

Hypothesis 1

In table 12 the results from running the first hypothesis on the dataset containing foot data from both healthy controls and patients with UPDRS 0.

Method	Cross-validated MSE ± SE	Tuning Parameter ± SE
Forward selection	0.188 ± 0.021	2.056 ± 0.056
Decision tree	0.197 ± 0.027	0.016 ± 0.003
K-nearest neighbor	0.066 ± 0.015	1.778 ± 0.461
Linear regression	0.465 ± 0.080	-
Local regression	0.669 ± 0.113	0.622 ± 0.046
Smoothing splines FS	0.256 ± 0.034	0.143 ± 0.014
MARS linear	0.118 ± 0.023	-
MARS cubic	0.126 ± 0.026	-
Neural networks	0.133 ± 0.024	-
Ridge regression	0.097 ± 0.022	35.556 ± 5.729
The lasso	0.084 ± 0.011	$0.010 \pm 4.207{*}10^{-19}$
SVM cubic	0.073 ± 0.028	$0.010 \pm 4.207{*}10^{-19}$
SVM poly	0.062 ± 0.012	$0.010 \pm 4.207{*}10^{-19}$
SVM linear	0.076 ± 0.012	0.091 ± 0.010
PCA decision tree	0.294 ± 0.029	$0.019 \pm 5.008{*}10^{-4}$
PCA K-nearest neighbor	0.255 ± 0.018	18.722 ± 1.576
PCA linear regression	0.114 ± 0.022	66.944 ± 4.032
PCA local regression	0.266 ± 0.019	0.781 ± 0.042
PCA neural network	0.293 ± 0.030	-
PCA smoothing splines	0.317 ± 0.028	0.178 ± 0.005

Table 12: Table displaying the resulting average cross-validated MSE together with the attached average standard error (SE), and the average tuning parameter with the attached average standard error. This table shows results for hypothesis 1 for foot data from Parkinson's disease patients with UPDRS 0 and healthy controls with UPDRS 0.

Hypothesis 2

In table 13 the results from running the second hypothesis on the dataset containing foot data from both healthy controls and patients with UPDRS 0.

Method	Cross-validated MSE ± SE	Tuning Parameter ± SE
Forward selection	0.285 ± 0.056	39.889 ± 6.426
Decision tree	0.170 ± 0.029	0.016 ± 0.002
K-nearest neighbor	0.077 ± 0.022	2.556 ± 0.246
Linear regression	0.151 ± 0.024	-
Local regression	0.208 ± 0.020	0.581 ± 0.048
Smoothing splines FS	0.157 ± 0.025	0.060 ± 0.016
MARS linear	0.129 ± 0.031	-
MARS cubic	0.147 ± 0.026	-
Neural networks	0.194 ± 0.022	-
Ridge regression	0.160 ± 0.024	5.000 ± 1.852
The lasso	0.168 ± 0.019	0.004 ± 0.001
SVM cubic	0.099 ± 0.024	$0.010 \pm 4.207{*}10^{-19}$
SVM poly	0.107 ± 0.017	$0.010 \pm 4.207{*}10^{-19}$
SVM linear	0.125 ± 0.025	0.167 ± 0.008
PCA decision tree	0.318 ± 0.034	0.020 ± 0.001
PCA K-nearest neighbor	0.247 ± 0.025	19.167 ± 1.373
PCA linear regression	0.207 ± 0.022	32.278 ± 4.726
PCA local regression	0.242 ± 0.030	0.539 ± 0.035
PCA neural network	0.264 ± 0.029	-
PCA smoothing splines	0.287 ± 0.027	0.179 ± 0.005

Table 13: Table displaying the resulting average cross-validated MSE together with the attached average standard error (SE), and the average tuning parameter with the attached average standard error. This table shows results for hypothesis 2 for foot data from Parkinson's disease patients with UPDRS 0 and healthy controls with UPDRS 0.

Hypothesis 3

In table 14 the results from running the third hypothesis on the dataset containing foot data from both healthy controls and patients with UPDRS 0.

Method	Cross-validated MSE ± SE	Tuning Parameter ± SE
Forward selection	0.209 ± 0.030	3.056 ± 0.098
Decision tree	0.132 ± 0.025	0.009 ± 0.001
K-nearest neighbor	0.142 ± 0.027	2.000 ± 0.840
Linear regression	0.166 ± 0.023	-
Local regression	0.190 ± 0.023	0.578 ± 0.044
Smoothing splines FS	0.164 ± 0.023	0.138 ± 0.012
MARS linear	0.154 ± 0.050	-
MARS cubic	0.142 ± 0.028	-
Neural networks	0.221 ± 0.026	-
Ridge regression	0.155 ± 0.025	0.000 ± 0.000
The lasso	0.202 ± 0.037	0.002 ± 0.001
SVM cubic	0.215 ± 0.031	0.166 ± 0.008
SVM poly	0.141 ± 0.019	0.030 ± 0.004
SVM linear	0.188 ± 0.024	0.119 ± 0.010
PCA decision tree	0.259 ± 0.022	0.012 ± 0.001
PCA K-nearest neighbor	0.242 ± 0.019	9.889 ± 0.767
PCA linear regression	0.215 ± 0.030	6.278 ± 0.158
PCA local regression	0.157 ± 0.024	0.756 ± 0.036
PCA neural network	0.266 ± 0.034	-
PCA smoothing splines	0.243 ± 0.025	0.160 ± 0.009

Table 14: Table displaying the resulting average cross-validated MSE together with the attached average standard error (SE), and the average tuning parameter with the attached average standard error. This table shows results for hypothesis 3 for foot data from Parkinson's disease patients with UPDRS 0 and healthy controls with UPDRS 0.

Hypothesis 4

In table 15 the results from running the fourth hypothesis on the dataset containing foot data from both healthy controls and patients with UPDRS 0.

Method	Cross-validated MSE ± SE	Tuning Parameter ± SE
Forward selection	0.198 ± 0.027	2.000 ± 0.013
Decision tree	0.186 ± 0.021	0.031 ± 0.004
K-nearest neighbor	0.138 ± 0.021	1.944 ± 0.274
Linear regression	0.241 ± 0.069	-
Local regression	0.857 ± 0.661	0.572 ± 0.040
Smoothing splines FS	0.185 ± 0.027	0.074 ± 0.015
MARS linear	0.113 ± 0.022	-
MARS cubic	0.144 ± 0.023	-
Neural networks	0.191 ± 0.026	-
Ridge regression	0.354 ± 0.221	12.778 ± 2.399
The lasso	0.139 ± 0.019	0.002 ± 0.001
SVM cubic	0.128 ± 0.027	$0.011 \pm 5.556{*}10^{-4}$
SVM poly	0.099 ± 0.020	$0.010 \pm 4.207{*}10^{-19}$
SVM linear	0.126 ± 0.018	0.151 ± 0.011
PCA decision tree	0.349 ± 0.034	$0.015 \pm 5.677{*}10^{-4}$
PCA K-nearest neighbor	0.282 ± 0.029	25.611 ± 1.732
PCA linear regression	0.240 ± 0.032	11.222 ± 3.667
PCA local regression	0.192 ± 0.063	0.406 ± 0.006
PCA neural network	0.265 ± 0.035	-
PCA smoothing splines	0.349 ± 0.025	0.171 ± 0.007

Table 15: Table displaying the resulting average cross-validated MSE together with the attached average standard error (SE), and the average tuning parameter with the attached average standard error. This table shows results for hypothesis 4 for foot data from Parkinson's disease patients with UPDRS 0 and healthy controls with UPDRS 0.

Hypothesis 5

In table 16 the results from running the fifth hypothesis on the dataset containing foot data from both healthy controls and patients with UPDRS 0.

Method	Cross-validated MSE ± SE	Tuning Parameter ± SE
Forward selection	0.186 ± 0.0300	2.944 ± 0.056
Decision tree	0.132 ± 0.024	0.015 ± 0.001
K-nearest neighbor	0.167 ± 0.034	11.778 ± 0.329
Linear regression	0.194 ± 0.024	-
Local regression	0.178 ± 0.021	0.442 ± 0.013
Smoothing splines FS	0.173 ± 0.028	0.070 ± 0.014
MARS linear	0.177 ± 0.036	-
MARS cubic	0.125 ± 0.020	-
Neural networks	0.269 ± 0.022	-
Ridge regression	0.166 ± 0.033	12.778 ± 2.778
The lasso	0.175 ± 0.032	0.007 ± 0.001
SVM cubic	0.147± 0.026	0.119 ± 0.013
SVM poly	0.166 ± 0.018	0.020 ± 0.003
SVM linear	0.178 ± 0.030	0.138 ± 0.009
PCA decision tree	0.261 ± 0.034	0.016 ± 0.002
PCA K-nearest neighbor	0.267 ± 0.029	17.000 ± 2.267
PCA linear regression	0.313 ± 0.036	15.222 ± 0.827
PCA local regression	0.272 ± 0.029	0.656 ± 0.057
PCA neural network	0.284 ± 0.032	-
PCA smoothing splines	0.259 ± 0.025	0.171 ± 0.007

Table 16: Table displaying the resulting average cross-validated MSE together with the attached average standard error (SE), and the average tuning parameter with the attached average standard error. This table shows results for hypothesis 5 for foot data from Parkinson's disease patients with UPDRS 0 and healthy controls with UPDRS 0.

Result plots hand data UPDRS 0 to UPDRS 3

The results from the third dataset and each hypothesis are below presented in tables.

Hypothesis 1

In table 17 the results from running the first hypothesis on the dataset containing hand data from Parkinson patients with UPDRS between 0 and 3.

Method	Cross-validated MSE ± SE	Tuning Parameter ± SE
Forward selection	1.251 ± 0.314	3.056 ± 0.654
Decision tree	1.146 ± 0.602	0.039 ± 0.003
K-nearest neighbor	0.758 ± 0.138	44.778 ± 0.979
Linear regression	6.102 ± 3.293	-
Local regression	4.292 ± 0.957	0.492 ± 0.025
Smoothing splines FS	0.905 ± 0.276	0.103 ± 0.033
MARS linear	1.277 ± 0.270	-
MARS cubic	1.046 ± 0.213	-
Neural networks	1.243 ± 0.210	-
Ridge regression	0.924 ± 0.240	392.778 ± 279.120
The lasso	0.729 ± 0.151	0.030 ± 0.013
SVM cubic	0.823 ± 0.142	0.010 ± $4.207*10^{-19}$
SVM poly	0.926 ± 0.200	0.010 ± $4.207*10^{-19}$
SVM linear	0.953 ± 0.172	0.142 ± 0.012
PCA decision tree	1.018 ± 0.653	0.035 ± 0.002
PCA K-nearest neighbor	1.063 ± 0.149	17.611 ± 4.315
PCA linear regression	0.803 ± 0.149	16.722 ± 5.884
PCA local regression	0.959± 0.199	0.481 ± 0.023
PCA neural network	1.209 ± 0.234	-
PCA smoothing splines	0.804 ± 0.125	0.158 ± 0.017

Table 17: Table displaying the resulting average cross-validated MSE together with the attached average standard error (SE), and the average tuning parameter with the attached average standard error. This table shows results for hypothesis 1 for hand data from Parkinson's disease patients.

Hypothesis 2

In table 18 the results from running the second hypothesis on the dataset containing hand data from Parkinson patients with UPDRS between 0 and 3.

Method	Cross-validated MSE ± SE	Tuning Parameter ± SE
Forward selection	1.430 ± 0.273	8.278 ± 0.235
Decision tree	1.096 ± 0.089	0.047 ± 0.001
K-nearest neighbor	0.734 ± 0.118	46.444 ± 0.868
Linear regression	0.938 ± 0.156	-
Local regression	1.095 ± 0.150	0.589 ± 0.047
Smoothing splines FS	0.642 ± 0.100	0.027 ± 0.010
MARS linear	0.930 ± 0.119	-
MARS cubic	0.901 ± 0.101	-
Neural networks	1.836 ± 0.368	-
Ridge regression	0.810 ± 0.135	447.778 ± 304.992
The lasso	0.852 ± 0.145	0.036 ± 0.020
SVM cubic	0.917 ± 0.164	0.010 ± $4.207*10^{-19}$
SVM poly	0.938 ± 0.146	0.010 ± $4.207*10^{-19}$
SVM linear	0.847 ± 0.157	0.132 ± 0.010
PCA decision tree	0.881 ± 0.128	0.035 ± 0.002
PCA K-nearest neighbor	0.843 ± 0.132	31.667 ± 3.757
PCA linear regression	0.730± 0.162	5.944 ± 3.075
PCA local regression	0.934 ± 0.206	0.903 ± 0.035
PCA neural network	2.134 ± 0.352	-
PCA smoothing splines	0.667 ± 0.093	0.149 ± 0.016

Table 18: Table displaying the resulting average cross-validated MSE together with the attached average standard error (SE), and the average tuning parameter with the attached average standard error. This table shows results for hypothesis 2 for hand data from Parkinson's disease patients.

Hypothesis 3

In table 19 the results from running the third hypothesis on the dataset containing hand data from Parkinson patients with UPDRS between 0 and 3.

Method	Cross-validated MSE ± SE	Tuning Parameter ± SE
Forward selection	1.455 ± 0.269	2.500 ± 0.167
Decision tree	0.946 ± 0.567	$0.048 \pm 4.951{*}10^{-4}$
K-nearest neighbor	0.607 ± 0.064	42.111 ± 1.682
Linear regression	0.653 ± 0.097	-
Local regression	0.751 ± 0.107	0.564 ± 0.039
Smoothing splines FS	1.037 ± 0.213	0.077 ± 0.013
MARS linear	0.890 ± 0.090	-
MARS cubic	0.731 ± 0.872	-
Neural networks	1.402 ± 0.373	-
Ridge regression	0.640 ± 0.099	150.556 ± 72.763
The lasso	0.632 ± 0.091	0.032 ± 0.013
SVM cubic	0.829 ± 0.102	0.164 ± 0.014
SVM poly	1.207 ± 0.173	0.036 ± 0.004
SVM linear	0.686 ± 0.129	0.076 ± 0.018
PCA decision tree	0.960 ± 0.101	0.044 ± 0.002
PCA K-nearest neighbor	0.719 ± 0.073	42.111 ± 2.840
PCA linear regression	1.070 ± 0.204	10.333 ± 0.767
PCA local regression	0.924 ± 0.152	0.714 ± 0.057
PCA neural network	1.692 ± 0.356	-
PCA smoothing splines	0.664 ± 0.067	0.146 ± 0.145

Table 19: Table displaying the resulting average cross-validated MSE together with the attached average standard error (SE), and the average tuning parameter with the attached average standard error. This table shows results for hypothesis 3 for hand data from Parkinson's disease patients.

Hypothesis 4

In table 20 the results from running the fourth hypothesis on the dataset containing hand data from Parkinson patients with UPDRS between 0 and 3.

Method	Cross-validated MSE ± SE	Tuning Parameter ± SE
Forward selection	1.392 ± 0.279	16.332 ± 3.270
Decision tree	0.903 ± 0.167	0.036 ± 0.002
K-nearest neighbor	0.768 ± 0.149	43.500 ± 1.679
Linear regression	1.701 ± 0.392	-
Local regression	2.424 ± 0.543	0.744 ± 0.051
Smoothing splines FS	1.952 ± 0.356	0.023 ± 0.011
MARS linear	0.811 ± 0.160	-
MARS cubic	1.002 ± 0.182	-
Neural networks	1.580 ± 0.416	-
Ridge regression	1.027 ± 0.201	595.000 ± 265.487
The lasso	0.887 ± 0.185	0.066 ± 0.082
SVM cubic	0.929 ± 0.150	0.010 ± $4.207*10^{-19}$
SVM poly	0.886 ± 0.177	0.010 ± $4.207*10^{-19}$
SVM linear	0.882 ± 0.170	0.145 ± 0.013
PCA decision tree	1.113 ± 0.175	0.044 ± 0.002
PCA K-nearest neighbor	0.971 ± 0.156	34.778 ± 4.334
PCA linear regression	1.340 ± 0.379	5.000 ± 0.511
PCA local regression	0.863 ± 0.141	0.522 ± 0.026
PCA neural network	2.255 ± 0.356	-
PCA smoothing splines	0.816 ± 0.118	0.143 ± 0.019

Table 20: Table displaying the resulting average cross-validated MSE together with the attached average standard error (SE), and the average tuning parameter with the attached average standard error. This table shows results for hypothesis 4 for hand data from Parkinson's disease patients.

Hypothesis 5

In table 21 the results from running the fifth hypothesis on the dataset containing hand data from Parkinson patients with UPDRS between 0 and 3.

Method	Cross-validated MSE ± SE	Tuning Parameter ± SE
Forward selection	1.098 ± 0.265	4.770 ± 0.218
Decision tree	1.207 ± 0.149	$0.046 \pm 8.394{*}10^{-4}$
K-nearest neighbor	0.726 ± 0.143	43.222 ± 2.276
Linear regression	0.927 ± 0.158	-
Local regression	0.969 ± 0.167	0.594 ± 0.046
Smoothing splines FS	0.606 ± 0.441	0.064 ± 0.016
MARS linear	0.833 ± 0.146	-
MARS cubic	0.786 ± 0.151	-
Neural networks	1.497 ± 0.321	-
Ridge regression	0.742 ± 0.151	523.333 ± 301.909
The lasso	0.692 ± 0.149	0.055 ± 0.019
SVM cubic	0.854 ± 0.1543	0.128 ± 0.012
SVM poly	0.973 ± 0.175	0.017 ± 0.002
SVM linear	0.874 ± 0.174	0.128 ± 0.015
PCA decision tree	1.074 ± 0.227	0.043 ± 0.001
PCA K-nearest neighbor	0.728 ± 0.177	279.722 ± 2.229
PCA linear regression	1.065 ± 0.258	5.667 ± 1.508
PCA local regression	1.061 ± 0.268	0.656 ± 0.0461
PCA neural network	1.239 ± 0.211	-
PCA smoothing splines	0.730 ± 0.150	0.135 ± 0.018

Table 21: Table displaying the resulting average cross-validated MSE together with the attached average standard error (SE), and the average tuning parameter with the attached average standard error. This table shows results for hypothesis 5 for hand data from Parkinson's disease patients.

Result plots for hand and foot data UPDRS 0 to UPDRS 3

The results from the fourth dataset and each hypothesis are below presented in tables.

Hypothesis 1

In table 22 the results from running the first hypothesis on the dataset containing hand data from Parkinson patients with UPDRS between 0 and 3.

Method	Cross-validated MSE ± SE	Tuning Parameter ± SE
Forward selection	1.064 ± 0.100	66.320 ± 5.901
Decision tree	0.638 ± 0.042	0.015 ± 0.004
K-nearest neighbor	0.524 ± 0.062	3.053 ± 0.609
Linear regression	0.721 ± 0.080	-
Local regression	0.781 ± 0.070	0.616 ± 0.045
Smoothing splines FS	1.088 ± 0.093	$0.002 \pm 9.407{*}10^{-4}$
MARS linear	0.502 ± 0.043	-
MARS cubic	0.470 ± 0.036	-
Neural networks	1.925 ± 0.208	-
Ridge regression	0.494 ± 0.027	47.368 ± 7.941
The lasso	0.504 ± 0.029	0.016 ± 0.001
SVM cubic	0.432 ± 0.032	$0.010 \pm 4.089{*}10^{-19}$
SVM poly	0.477 ± 0.041	$0.010 \pm 4.089{*}10^{-19}$
SVM linear	0.608 ± 0.046	0.055 ± 0.008
PCA decision tree	1.265 ± 0.100	0.023 ± 0.004
PCA K-nearest neighbor	1.031 ± 0.077	13.368 ± 1.021
PCA linear regression	0.669 ± 0.045	82.790 ± 2.145
PCA local regression	0.771 ± 0.055	$0.400 \pm 2.617{*}10^{-17}$
PCA neural network	2.441 ± 0.279	-
PCA smoothing splines	0.721 ± 0.053	$0.003 \pm 2.105{*}10^{-4}$

Table 22: Table displaying the resulting average cross-validated MSE together with the attached average standard error (SE), and the average tuning parameter with the attached average standard error. This table shows results for hypothesis 1 for both foot and hand data from Parkinson's disease patients.

Hypothesis 2

In table 23 the results from running the second hypothesis on the dataset containing hand data from Parkinson patients with UPDRS between 0 and 3.

Method	Cross-validated MSE ± SE	Tuning Parameter ± SE
Forward selection	0.936 ± 0.080	31.158 ± 3.954
Decision tree	0.770 ± 0.098	0.015 ± 0.003
K-nearest neighbor	0.531 ± 0.048	5.744 ± 0.338
Linear regression	0.676 ± 0.046	-
Local regression	0.743 ± 0.056	0.474 ± 0.017
Smoothing splines FS	1.104 ± 0.090	0.030 ± 0.009
MARS linear	0.547 ± 0.038	-
MARS cubic	0.578 ± 0.045	-
Neural networks	1.710 ± 0.181	-
Ridge regression	0.608 ± 0.032	163.684 ± 11.061
The lasso	0.616 ± 0.044	0.016 ± 0.002
SVM cubic	0.502 ± 0.039	0.015 ± 0.001
SVM poly	0.614 ± 0.059	$0.010 \pm 4.089{*}10^{-19}$
SVM linear	0.692 ± 0.052	0.070 ± 0.009
PCA decision tree	1.359 ± 0.090	0.030 ± 0.003
PCA K-nearest neighbor	1.083 ± 0.076	22.947 ± 1.713
PCA linear regression	0.708 ± 0.054	66.737 ± 1.184
PCA local regression	0.832 ± 0.042	0.416 ± 0.013
PCA neural network	2.822 ± 0.163	-
PCA smoothing splines	0.801 ± 0.054	0.007 ± 0.001

Table 23: Table displaying the resulting average cross-validated MSE together with the attached average standard error (SE), and the average tuning parameter with the attached average standard error. This table shows results for hypothesis 2 for both foot and hand data from Parkinson's disease patients.

Hypothesis 3

In table 24 the results from running the third hypothesis on the dataset containing hand data from Parkinson patients with UPDRS between 0 and 3.

Method	Cross-validated MSE ± SE	Tuning Parameter ± SE
Forward selection	1.106 ± 0.090	1.526 ± 0.526
Decision tree	0.717 ± 0.041	0.010 ± $8.946*10^{-4}$
K-nearest neighbor	0.665 ± 0.056	7.790 ± 1.768
Linear regression	0.725 ± 0.033	-
Local regression	0.642 ± 0.045	0.408 ± 0.008
Smoothing splines FS	1.020 ± 0.081	0.013 ± 0.006
MARS linear	0.537 ± 0.043	-
MARS cubic	0.580 ± 0.038	-
Neural networks	1.985 ± 0.223	-
Ridge regression	0.692 ± 0.027	97.895 ± 15.094
The lasso	0.666 ± 0.033	0.051 ± 0.004
SVM cubic	0.596 ± 0.064	0.026 ± 0.007
SVM poly	0.587 ± 0.040	0.013 ± 0.001
SVM linear	0.743 ± 0.063	0.128 ± 0.012
PCA decision tree	1.365 ± 0.119	0.020 ± $8.606*10^{-4}$
PCA K-nearest neighbor	1.093 ± 0.067	9.368 ± 2.949
PCA linear regression	0.742 ± 0.053	19.543 ± 2.670
PCA local regression	0.930 ± 0.060	0.682 ± 0.046
PCA neural network	2.172 ± 0.276	-
PCA smoothing splines	0.848 ± 0.063	0.004 ± 0.001

Table 24: Table displaying the resulting average cross-validated MSE together with the attached average standard error (SE), and the average tuning parameter with the attached average standard error. This table shows results for hypothesis 3 for both foot and hand data from Parkinson's disease patients.

Hypothesis 4

In table 25 the results from running the fourth hypothesis on the dataset containing hand data from Parkinson patients with UPDRS between 0 and 3.

Method	Cross-validated MSE ± SE	Tuning Parameter ± SE
Forward selection	1.207 ± 0.053	24.683 ± 3.201
Decision tree	0.839 ± 0.070	0.012 ± 0.001
K-nearest neighbor	0.602 ± 0.062	2.895 ± 0.602
Linear regression	1.024 ± 0.416	-
Local regression	0.748 ± 0.118	0.408 ± 0.006
Smoothing splines FS	0.976 ± 0.086	0.136 ± 0.011
MARS linear	0.608 ± 0.044	-
MARS cubic	0.615 ± 0.056	-
Neural networks	1.911 ± 0.180	-
Ridge regression	0.625 ± 0.038	76.842 ± 11.447
The lasso	0.618 ± 0.039	$0.010 \pm 7.647{*}10^{-4}$
SVM cubic	0.546 ± 0.054	0.016 ± 0.002
SVM poly	0.520 ± 0.045	$0.010 \pm 4.089{*}10^{-19}$
SVM linear	0.668 ± 0.067	0.117 ± 0.013
PCA decision tree	1.367 ± 0.115	0.015 ± 0.003
PCA K-nearest neighbor	1.074 ± 0.064	36.632 ± 1.867
PCA linear regression	0.704 ± 0.035	73.526 ± 0.345
PCA local regression	0.794 ± 0.051	0.416 ± 0.006
PCA neural network	2.977 ± 0.170	-
PCA smoothing splines	0.748 ± 0.069	0.033 ± 0.004

Table 25: Table displaying the resulting average cross-validated MSE together with the attached average standard error (SE), and the average tuning parameter with the attached average standard error. This table shows results for hypothesis 4 for both foot and hand data from Parkinson's disease patients.

Hypothesis 5

In table 26 the results from running the fifth hypothesis on the dataset containing hand data from Parkinson patients with UPDRS between 0 and 3.

Method	**Cross-validated MSE ± SE**	**Tuning Parameter ± SE**
Forward selection	0.980 ± 0.094	9.368 ± 1.245
Decision tree	0.832 ± 0.081	0.017 ± 0.004
K-nearest neighbor	0.784 ± 0.061	4.842 ± 0.659
Linear regression	0.778 ± 0.046	-
Local regression	0.745 ± 0.046	0.426 ± 0.019
Smoothing splines FS	1.112 ± 0.094	0.132 ± 0.007
MARS linear	0.624 ± 0.053	-
MARS cubic	0.601 ± 0.043	-
Neural networks	1.771± 0.224	-
Ridge regression	0.790 ± 0.050	20.526 ± 3.705
The lasso	0.780 ± 0.046	0.001 ± $7.234*10^{-4}$
SVM cubic	0.587 ± 0.061	0.046 ± 0.012
SVM poly	0.661 ± 0.067	0.018 ± 0.006
SVM linear	0.797 ± 0.060	0.086 ± 0.011
PCA decision tree	1.143 ± 0.087	0.013 ± 0.003
PCA K-nearest neighbor	1.027 ± 0.076	44.368 ± 1.207
PCA linear regression	0.800 ± 0.063	15.790 ± 0.847
PCA local regression	0.957 ± 0.069	0.540 ± 0.024
PCA neural network	2.319 ± 0.298	-
PCA smoothing splines	0.834 ± 0.048	0.013 ± 0.002

Table 26: Table displaying the resulting average cross-validated MSE together with the attached average standard error (SE), and the average tuning parameter with the attached average standard error. This table shows results for hypothesis 5 for both foot and hand data from Parkinson's disease patients.