

École polytechnique de Louvain

Predictive models of Parkinson disease from movement data

Authors: **Lionel OVAERT, Noémie VERSTRAETE**
Supervisors: **Pierre DUPONT, Guillaume BERNARD**
Readers: **Alexander GERNIERS, Jean VANDERDONCKT**
Academic year 2019–2020
Master [120] in Computer Science
Master [120] in Computer Science and Engineering

Abstract

The Movement Disorder Society - Unified Parkinson's Disease Rating Scale (MDS-UPDRS) is a scale used by physicians to assess the evolution of the Parkinson's Disease (PD) in their patients. However, this assessment suffers from physician bias and variability. The biotech Tools4Patient, partner of this thesis, has developed the SensorMotor, a tool that records acceleration and angular velocity of specific hand movements of the MDS-UPDRS in order to predict more accurate and less variable scores to assess the progression of the disease.

The first contribution of this thesis is to propose machine learning approaches to analyze these movement data. The second is to study the influence of the age and of the training effect on the recorded movements by using the proposed machine learning models to predict the age and the repetition number of a record. Needed steps to achieve these goals are to acquire several repetitions of good quality movement data through the development of a protocol, and to propose features extraction techniques to build an informative data set for the proposed machine learning models: Support Vector Machine (SVM) and Multi-Layer Perceptron (MLP). Eventually, under the assumption that the training effect would be quite small, the following idea is proposed: summarize the data for the different repetitions of a person's movement by keeping only the median value for each feature of the data set, in order to improve the age range predictions.

Results show that the influence of the age and the training effect are quite low, but nevertheless present in the recorded movements. Moreover, the comparison of the performances of the different machine learning models shows that the use of the proposed summarized data allows to statistically improve the age range prediction of the best ranked classifier for movement data.

Acknowledgments

First of all, we would like to thank our supervisors, Professor Pierre Dupont and Guillaume Bernard, for their relevant comments, their availability, and their precious advice both on the process, the form and the substance of our thesis.

Thanks also to Barbara Pachikian and the CICN, who lent us a room and gave us advice for the data collection.

Finally, a big thank you to the staff of Tools4Patient as well as to our families, friends and acquaintances who participated in the data collection.

Contents

Abstract	i
Acknowledgements	ii
Acronyms	vi
List of Figures	vii
List of Tables	1
Introduction	2
1 Background	5
1.1 MDS-UPDRS	5
1.2 Description of SensorMotors	7
1.3 State of the art	8
1.3.1 Effect of age on motor speed	8
1.3.2 Finger tapping: evaluation of useful parameters	8
1.3.3 Tremors and useful parameters	9
2 Data	11
2.1 Protocol	11
2.1.1 Inclusion criteria	11
2.1.2 Context	12
2.1.3 Informed consent	12
2.2 Recording the data	13
2.2.1 Finger tapping & pronation-supination	13
2.2.2 Postural tremor & kinetic tremor	14
2.3 Overview of the participants according to their age, gender and dominant hand	15
3 Solution overview	17

4	Preprocessing	19
4.1	Remove the hand symmetry	19
4.2	Finger tapping and pronation-supination	21
4.2.1	Segment the signals	21
4.3	Postural and kinetic tremors tasks	26
4.3.1	Segment the signals	26
4.3.2	Preprocessing acceleration data	27
4.3.3	Double integration of acceleration data	28
4.3.4	Power Spectral Density	28
5	Feature Extraction	31
5.1	Features based on movement	31
5.1.1	Accelerometer-specific features	32
5.1.2	Gyroscope-specific features	33
5.1.3	Non-specific features	34
5.2	Features based on tremors	35
5.2.1	Features extracted from accelerations and angular velocities	35
5.2.2	Features extracted from positions	36
5.2.3	Features extracted from power spectral density	37
6	Data visualization and dimensionality reduction	39
6.1	Relevance and redundancy of the features	39
6.1.1	Relevance	40
6.1.2	Redundancy	41
6.2	Principal Component Analysis and dimensionality reduction	43
6.3	Visualization of the two first principal components	45
6.4	Reduce the noise for the age prediction task	46
7	Discriminative models for classification tasks	49
7.1	Formalisation of the supervised classification tasks	49
7.2	Support Vector Machine	50
7.3	Multi-layer Perceptron	54
7.3.1	Structure	54
7.3.2	Learning	56
7.3.3	Advantages and disadvantages of MLP	58
8	Multi-dataset and multi-class performance assessment	59
8.1	Hold-out testing	59
8.1.1	Data leakage	60
8.1.2	Data imbalance	60
8.1.3	Data resampling	61

8.2	Cross-validation	61
8.3	Splitting with attention to groups and class balance for the age range prediction	61
8.4	Performance metrics	62
8.5	Multi-classifiers and multi-dataset performance assessment	63
8.5.1	Friedman test	63
8.5.2	Post-hoc tests	64
9	Experimental results	66
9.1	Data sets	66
9.2	Experimental settings	67
9.3	Model hyperparameters	68
9.4	Comparison of classifiers performances	69
9.4.1	Overall comparison of classifiers	70
9.4.2	Comparison of the classifiers for the repetition number prediction task	74
9.4.3	Comparison of the classifiers for the age range prediction task	75
9.5	Denoised data for the age range prediction	77
9.6	Discussion	78
	Conclusion	81
	Bibliography	83
A	Results of data set resamples	87

Acronyms

FT Finger tapping.

KT Kinetic tremor of the hands.

MDS-UPDRS Movement Disorder Society - Unified Parkinson's Disease Rating Scale.

MLP Multi-Layer Perceptron.

PS Pronation-Supination movements of hand.

PT Postural tremor of the hands.

SV Support Vectors.

SVM Support Vector Machine.

List of Figures

1.1	Bluetooth module (A) attached around the arm using provided strap; Finger sensors (B) attached around the fingers using silicon rubber bands	7
2.1	Left : Finger Tapping. Right : Pronation-supination movements of hands. Both images are provided by Tools4Patient.	14
2.2	Left : Postural tremor of the hands. Right : Kinetic tremor of the hands. Both images are provided by Tools4Patient.	14
2.3	Non-uniform age distribution.	15
2.4	3 unbalanced subgroups for the age prediction. Both men and women are present in each group.	16
4.1	Axes orientation of accelerometers and gyroscopes	20
6.1	30 features with the highest correlation coefficient w.r.t the repetition number (top) and the age (bottom).	41
6.2	Heatmap of the correation coefficient matrix. The blue, yellow, green and purple squares represent the correlations of features extracted from FT, PS, PT and KT tasks respectively. Pink squares represent the correlations between FT and PS tasks.	43
6.3	First principal component according to the second with colors representing repetition number (left) and age (right).	45
6.4	First principal component according to the second with colors representing repetition number (left) and age (right). In these graphs, outliers (samples where the value of at least one of the two principal components is not between the 5th and 95th percentiles) have been removed.	46
6.5	30 features with the highest coefficient correlation w.r.t the age for the "denoised" data.	47
6.6	Heatmap of the correation coefficient matrix for the "denoised" data.	48
6.7	First principal component according to the second with colors representing the age for the "denoised" data.	48

7.1	2D examples of hyperplanes. Left: Multiple separating hyperplanes L_i with different margins. Right: Maximal margin hyperplane with highlighted support vectors.[25]	50
7.2	Structure of a multi-layer perceptron : one input layer, at least one hidden layer (two in this example) and one output layer.	55
9.1	Comparison of all classifiers against each other using the Nemenyi test. Classifiers belong to the same group (connected with a black line) and their performances are not significantly different at $\alpha = 0.05$.	72
9.2	Comparison of all classifiers against each other using the Nemenyi test. Classifiers that belong to the same group (connected with a black line) are not significantly different at $\alpha = 0.1$.	72
9.3	Comparison of a random classifier against the other classifiers using the Bonferroni-Dunn test. All classifiers whose rank is outside the marked interval are significantly different from the baseline random classifier at $\alpha = 0.1$.	73
9.4	Comparison of a random classifier against the other classifiers using the Bonferroni-Dunn test. All classifiers whose rank is outside the marked interval are significantly different from the baseline random classifier at $\alpha = 0.05$.	73
9.5	Comparison of all classifiers against each other using the Nemenyi test for the repetition number prediction task. Classifiers belonging to the same group (connected with a black line) do not have performances significantly different at $\alpha = 0.05$.	75
9.6	Comparison of all classifiers against each other using the Nemenyi test for the age range prediction task. Classifiers belonging to the same group (connected with a black line) do not have performances significantly different at $\alpha = 0.05$.	76
9.7	Comparison of all classifiers against each other using the Nemenyi test for the age range prediction task and with the denoised data set. Classifiers belonging to the same group (connected with a black line) do not have performances significantly different at $\alpha = 0.05$.	77

List of Tables

1.1	Precision of the 3-axis accelerometers and 3-axis gyroscopes	8
7.1	Comparison of some of the most popular activation functions. Logistic and hyperbolic tangent functions were the most used before the emergence of the Rectified Linear Unit (ReLU) function, which is now considered as the best default choice.	56
8.1	Train-test split with attention to groups and class balance	62
8.2	Critical values for the two-tailed Nemenyi test	64
8.3	Critical values for the two-tailed Bonferroni-Dunn test	65
9.1	Comparison of balanced accuracy scores between a random classifier, SVM with default Hyperparameters (HP), SVM tuned for linear and rbf kernel, MLP with default HP and MLP tuned for tanh and relu activation functions. The numbers in parentheses represent the ranks that are computed for the Friedman test.	70
9.2	Critical values for the two-tailed Nemenyi test	71
9.3	Critical values for the two-tailed Bonferroni-Dunn test	73
A.1	Age prediction results for 200 resamples with ranks.	87
A.2	Age denoised prediction results for 200 resamples with ranks.	97
A.3	Repetition number prediction results for 200 resamples with ranks.	107

Introduction

The Parkinson's disease (PD) is a neurodegenerative disorder of the nervous system where neurons, mainly those that produce dopamine, are damaged or die. According to the Parkinson's Foundation, more than 10 million people worldwide are living with PD[24]. The causes are unknown but there are some risk factors like age, sex, or heredity. The principal symptoms are the following: tremor, rigidity, bradykinesia (slowing down of spontaneous and automatic movement) as well as postural instability, and there is no cure even if some medication or surgery exists to assuage the symptoms of the disease[20].

In order to evaluate some aspects of PD, the Movement Disorder Society (MDS) has created the MDS-UPDRS. This scale is used by physicians as well as in research and includes four parts. One of them focuses on motor examination and consists of 18 tasks[17]. Physicians evaluate the tasks visually and give the patient a score from 0 to 4 for each task.

In this context, the biotech Tools4Patient, the partner of this thesis, tries to better characterize the movement disorders associated with PD. For this purpose, they have developed the Sensormotor, a device designed to record acceleration and angular velocities for specific movements of the MDS-UPDRS. Since today's tests are subject to clinician bias and variability, with the addition of this device, the score of a patient could be determined with higher precision and data could also be stored to be re-analysed later.

The collected data could however suffer from certain biases, for example the patient's age or a training effect if the movement has to be repeated for some reason. This is where this thesis comes in. Data is acquired using the SensorMotors from four specific tasks of the third part of the MDS-UPDRS scale. Each task is repeated a total of three times for each hand. Then, this thesis focuses on two possible biases that could have an influence on the movement data recorded by the SensorMotors : the effect of the age on the movements and the training effect when a movement is repeated several times.

The goal is to answer the following questions :

- What are the best machine learning approaches to analyze these movement data ?
- What is the influence of the age on the quality of these movements ?
- How is the training effect marked on these movements ?

The work for this thesis has therefore been divided into three main steps : firstly, acquire movement data from the healthy volunteers who have been recruited, secondly, propose efficient approaches to preprocess the recorded data and finally, analyse and build machine learning models to predict both an age range and the repetition number of the recorded movements.

The first chapter describes the general framework of the work : a general introduction to MDS-UPDRS, a description of the four tasks on which this thesis focuses, and a state of the art on some aspects related to the thesis.

Then, the protocol used to collect the data for the four movements tasks is described in the second chapter, and a general overview of the data of the volunteers that participated to the study is given.

The third chapter presents a general overview of proposed solution described in the following chapters by a pipeline representing the most important steps.

Afterwards, the preprocessing steps on the records before extracting useful features are depicted in the fourth chapter, while the fifth chapter focuses on this features extraction to constitute a data set.

The next chapter examines the collected data by using the Pearson's coefficient correlation and the first two PCA components in order to gain some intuition about which machine learning models might be appropriate. In the same way, PCA is chosen as a dimensionality reduction technique and some justifications are given. At the end of the chapter, we propose, based on the observations, to build a "denoised" version of the data set for the age range prediction.

The seventh chapter focuses on the theoretical aspects of the two chosen discriminative classifiers : the Support Vector Machine (SVM) and the Muti-Layer Perceptron (MLP).

Before running the classifiers, the way the performances are assessed and the different techniques used to allow for a correct performance assessment of the results are described. Chapter eight explains this, while chapter nine presents the experimental results obtained with these performance assessment techniques and discusses the results.

In the final chapter, conclusions are drawn, results are summarized and further improvements are discussed.

Chapter 1

Background

This chapter presents the background concepts behind the study. This allows the reader to understand the movements in the MDS-UPDRS as well as the tools used. An overview of articles related to the study is also given.

1.1 MDS-UPDRS

Since 2008, doctors have assessed the evolution of the Parkinson Disease with a scale called MDS-UPDRS[2]. MDS-UPDRS is composed of four main parts each divided into smaller sub-parts. Each part aims at rating a series of exercises linked to PD. The first part covers the non-motor experiences of daily living. The second part assesses the motor experiences of the daily living. The third part is a motor examination and the fourth part covers the possible motor complications.

While all parts are important to rate the evolution of PD, this thesis focuses on four specific tasks of the third part. That portion of the scale assesses the motor signs of PD. Eighteen different tests are performed covering multiple parts of the body that are affected by PD such as the face, the hands, the legs but also the general posture. Each test is evaluated according to its own criteria and the results are given on a scale from 0 to 4.

The 5 scores are organised like this:

- 0: Normal
- 1: Slight
- 2: Mild
- 3: Moderate
- 4: Severe

This thesis highlights 4 exercises: Finger tapping (FT), Pronation-Supination movements of hand (PS), Postural tremor of the hands (PT) and Kinetic tremor of the hands (KT). In order to understand these tasks, we describe them in more detail below.

Finger Tapping

The first exercise is the tapping of the index and the thumb. The aim of this exercise is speed while opening the fingers as wide as possible. Each hand is rated separately and the evaluation done by the clinicians covers speed, amplitude, hesitations, halts and decrementing amplitude. This movement is performed 10 times by the patient.

Pronation-Supination movements of hands

For the second exercise, the patient has to hold his arm outstretched in front of him with his palm pointed to the ground. He then has to perform 180° rotations with his hand in order to have his palm pointing as much as possible to the sky. As for the FT exercise, the aim is to do this exercise as extensively and as fast as possible and the same components are evaluated. The movement is done 10 times.

Postural Tremor of the hands

In this third exercise, the posture of the hand is the same for each patient, arm outstretched in front of him with the palm pointing to the ground. The aim of this exercise is to hold hands as still as possible. This exercise is used to detect rest tremor in the hands of the patient. In this movement, all kinds of tremor are rated. One kind of tremor specific to Parkinson Disease is the re-emergent tremor. It represents a rest tremor that re-emerges after a variable delay while maintaining posture[13]. For each tremor, the highest amplitude seen is used as rating. This task has to be performed for 10 seconds.

Kinetic Tremor of the hands

In this fourth exercise, the investigator holds up a finger and the patient performs round trips with his finger between his nose and the finger of the investigator. This exercise should be performed as slowly as possible in order to avoid having fast arm movements hiding any kind of tremor. The tremor can be present either as the patient reaches one of the targets (nose or finger) or throughout the whole movement. As for PT, the highest amplitude of each tremor is used as rating. The movement is performed for each hand separately.

1.2 Description of SensorMotors

Using the two SensorMotors provided by Tools4Patient, one for each hand, the data has been acquired for the four specific movements detailed above. This section briefly describes the SensorMotor device and the data acquired with it.



Figure 1.1: Bluetooth module (A) attached around the arm using provided strap; Finger sensors (B) attached around the fingers using silicon rubber bands

As visible in Figure 1.1, a SensorMotor is composed of [22, 26]¹ :

- **two finger sensors**, one that is strapped at the tip of the thumb, and the other on the tip on the index using silicon rubber bands.
- **a Bluetooth module** (also composed of a battery and a microcontroller board) attached to the arm of the volunteers.
- **two wires cable** to connect each finger sensor with the Bluetooth module.

The finger sensors are IMU sensors. Their integrated 3-axis accelerometers and 3-axis gyroscopes allow to record acceleration [g] and angular velocities [°/sec]

¹The image and its caption were taken from [22], and the information comes from [22, 26]

data at a frequency sampling of 250Hz. Recording of each movement task is controlled with an application installed on a tablet connected by Bluetooth to the SensorMotors. Both the tablet and the application are provided by Tools4Patient. The precision of the accelerometers and gyroscopes of each finger sensor are set for each movement task as showed on Table 1.1.

Task \ Precision	accelerometer [g]	gyroscope [°/sec]
Finger Tapping	± 16	± 2000
Pronation-Supination	± 8	± 2000
Postural tremors	± 2	± 250
Action tremors	± 8	± 1000

Table 1.1: Precision of the 3-axis accelerometers and 3-axis gyroscopes

1.3 State of the art

1.3.1 Effect of age on motor speed

Based on the study of Ruff and Parker[21], several assumptions can be made regarding the effect of age, gender and handedness on the FT test. Experiments made for both the dominant and non-dominant hands have shown that the effect of age on motor speed is in fact moderated by gender. In other words, whereas men don't show any sign of decrease in performance in FT speeds with increasing age, women's performances decrease with increasing age.

Similar results are expected on the PS task given that this exercise also focuses on motor speed.

1.3.2 Finger tapping: evaluation of useful parameters

According to Yokoe et al.[29] and Stamatakis et al.[23], specific parameters could help in the process of evaluating the evolution of the Parkinson's disease in patients. In their relative studies, they have both used accelerometers that were fixed at the end of the index and the thumb in order to record acceleration data during the FT test. Both studies have recruited PD patients as well as healthy volunteers to compare results between these two study groups.

The team of Yokoe et al. have found that parameters related to velocity and amplitude give the best results to classify all the patients in their corresponding class. Out of their study, fourteen parameters are classified into three components

using principal component analysis: (1) mean and standard deviation of both amplitude and velocity, (2) number of finger tapping for 60 seconds and mean finger tapping interval, and (3) standard deviation of finger tapping interval. Among all these parameters, maximum opening velocity is found to be the best discriminating parameter.

Stamatakis et al. have focused on accelerometer-specific and MDS-UPDRS-specific features. Based on the computed movement frequency, percentage of movement time for maximum opening acceleration, maximum closing and opening acceleration, opening angle and level of hypometria they have extracted 18 different features. From these features, they have identified the indices of ascending and descending frequency, hesitations and halts, and slopes of time of maximum opening acceleration and maximum closing acceleration to be the best features to predict the MDS-UPDRS score of a patient.

1.3.3 Tremors and useful parameters

Tremors expected in the study

The tremor is defined by MDS as a *rhythmic oscillatory movement of a body part*[8]. In this thesis, the tremors of the hand are the most important. Among the four tasks of MDS-UPRS studied in this thesis, two tasks are related to specific types of tremors which occur for people with Parkinson’s disease : the tasks *postural tremor of the hands* (PT), related to postural tremors occurring when people maintain voluntarily a position against gravity, and *kinetic tremors of the hands* (KT), related to kinetic tremors that occur during a voluntary movement. These tremors, when they are observed for people with the Parkinson’s disease, have specific characteristics, and the principal one is the frequency range[11].

It is important to note that the volunteers from whom the data is collected are healthy. Therefore, they are not expected to have these kind of tremors with this specific frequency range, but only physiological tremors usually with high frequency and low amplitude that can be enhanced by anxiety, stress, fatigue and medication[3, 11]. Caffeine or physical activities could also enhance these tremors.

All the tremors (pathological or non pathological) can however be characterized by their amplitude and frequency, so techniques to extract informative features from Parkinson’s disease tremors could also be used for the physiological tremors, and vice-versa.

Signal processing and features extraction to characterize tremors

Several techniques and types of analysis (time-domain analysis, spectral analysis, time-frequency analysis, etc.) exist to extract informative features to characterize tremors from IMU data. Some of them are briefly mentioned here and are the ones on which the features extraction part is based.

On the one hand, for the time-domain, the Root mean square (RMS) of the acceleration and angular velocities[16, 6] can be used, which indicates the vibration level. Displacements can also be computed from acceleration data by integrating them twice and extract several features such as the mean amplitude and the standard deviation[14].

On the other hand, for the frequency domain, several studies use a spectral estimation such as the power spectral density (PSD) to extract features like the tremor frequency, considered as the dominant frequency of the PSD, or the tremor amplitude which is the area under the curve[11, 6].

Chapter 2

Data

This chapter describes and analyses the protocol that has been defined to collect data. A general overview of the participants is also given.

2.1 Protocol

An end-to-end procedure was defined to collect the data. In this protocol, we defined inclusion criteria, a general context to explain our study and set up an informed consent form with the help of Tools4Patient and the CICN¹. Every volunteer was given the same explanation in the same way.

2.1.1 Inclusion criteria

In order to be a participant of the study, some basic participation criteria has to be met. The participant has to be healthy, which is characterized by the fact that this person does not suffer from any upper limb movement disorder nor any neurological disorder in order to avoid some important bias in the data. An age limit is used as a way to define the part of the population that is going to be used in the study. The age limit is fixed at 18 for the minimum value. No age limit is set for the maximum value as long as the volunteers are able to perform the basic movements required for the study.

¹Centre d'Investigation Clinique en Nutrition

2.1.2 Context

The first step is to inform the participants in detail. The context of the study is set to make the academic insight as well as the final objectives clear. A basic introduction of MDS-UPDRS is added in order to inform the participants that all the movements they are going to perform are based on existing medical documents. Participants are also informed about the partnership with Tools4Patient and their involvement in this study.

The context also covers which data is collected and how. The set of personal information is defined in order to better characterize the data collected from the volunteers. This set consists of the age, the sex and the dominant hand. Collecting the age of the volunteer is mandatory for the training of the future age prediction models whereas the sex and the dominant hand would serve as features in the analysis as those could have an influence on the data[21]. For the prediction of the repetitions, the collected information serves as features. These information also allow to understand the sample of the population. The volunteers have to write whether they are male or female and for the dominant hand, the choice between left-handed, right-handed and ambidextrous is given as it was anticipated that the sample would contain one or more ambidextrous volunteers. As a final note, if volunteers have any question related to the study, an informative is provided to them.

2.1.3 Informed consent

The last step before proceeding to the real recording involves signing an informed consent form. This form gives a small summary of the before-mentioned context, inclusion criteria and collected data. In addition to all the previously mentioned information, this form contains a small section about the anonymity of the data. Once someone registered as a participant, only the data collected from the participant's movements as well as his/her age, sex and dominant hand would be used in the study. The form serves as a proof that the data is indeed collected from real people and each person signs it without any constraint. It also allows the participants to withdraw their participation in the case they don't want to take part in the study anymore. A last section covers how they could ask for the retrieval of their data from the study. Every volunteer is informed about this option.

2.2 Recording the data

Two tablets and two SensorMotors are provided by Tool4Patient to carry out the recordings. The second tablet is mainly present for when the first one would run out of power. These tablets are configured to have the recording application in the form of an Android application. When loading up the application, the SensorMotors connect through Bluetooth to the tablets.

Before starting to record, the volunteers are given the informed consent form. Once they are fully aware of the procedure, they are given a small form to fill in the personal information needed to perform some analysis (age, sex and dominant hand). An explanation on which part of the body the SensorMotors would be installed on is given before proceeding to the installation according to the settings defined by the tablet. Once everything is checked and properly installed, one complete iteration is performed.

Participants are informed that they can start performing a task once they hear the first beep sound and to stop once they hear the second beep. This sound is produced by the tablet when the start or stop button is hit. For each task, while the participants are making the required movements, the number of times he/she performs the movements is counted visually and the test is stopped once they have done three to five more movements than the ones needed. This also avoids that participants would slow down if they know they only have to perform a fixed amount of movements. Three series of four tasks are recorded in one session. Each task is thus performed before moving to the next series. Each hand is recorded separately for each task except for postural tremor where both hands are recorded jointly.

2.2.1 Finger tapping & pronation-supination

For finger tapping, participants are asked to hold their hand in front of them with their index vertically aligned. The movement is demonstrated and the participants are asked to perform this movement as widely and as fast as possible but to keep in mind to do the movement the right way. For the pronation-supination task, participants have to hold their hand stretched out in front of them with the palm of their hand pointing to the ground. Their fingers have to be slightly apart from each other. As for finger tapping, the main goal is speed and amplitude with an eye on correctness.

An example of the FT and PS movements can be found on Figure 2.1.

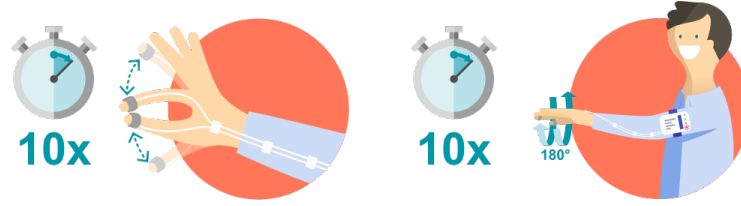


Figure 2.1: Left : Finger Tapping. Right : Pronation-supination movements of hands. Both images are provided by Tools4Patient.

2.2.2 Postural tremor & kinetic tremor

The postural and action tremor tasks are different whereas the main goal is to be accurate. For postural tremor, the same pose as during pronation supination is asked but both hands are recorded together. For kinetic tremor, the movement is demonstrated by emphasising that this task is about accuracy. The investigator has to insist on the fact that during the movement, participants have to rotate their finger between the moment they touched our finger and the moment they touched their nose. The PT and KT movements are demonstrated on Figure 2.2.

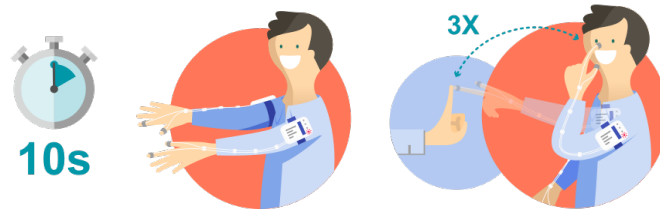


Figure 2.2: Left : Postural tremor of the hands. Right : Kinetic tremor of the hands. Both images are provided by Tools4Patient.

2.3 Overview of the participants according to their age, gender and dominant hand

A total of 59 volunteers between 19 and 87 years old have taken part in the study. In this subsection, these participants are briefly characterized with the collected personal information.

Age. The distribution of the volunteers according to their age is shown on Figure 2.3 as a bar chart. A first observation allows to notice the uneven distribution of ages within the sample. The median age is 31 years. For the prediction task of the age, we have divided the total age range (19 to 87 years) into three smaller age ranges that will be the categories to predict : 19-41, 42-64, 65-87 years. In this way, all ranges have the same length. An important point to see is that the age categories are not equally balanced.

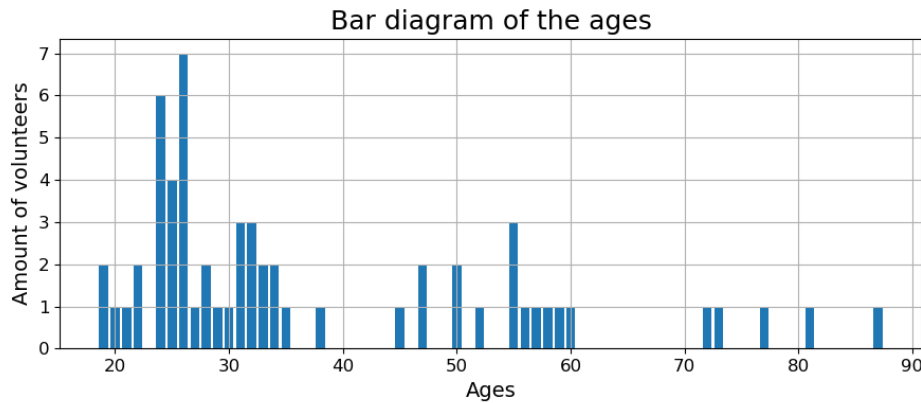


Figure 2.3: Non-uniform age distribution.

The number of participants for each age range is shown on Figure 2.4 : 40 volunteers for the first age range, 14 for the second and 5 participants for the third group. Thenceforth, we can hypothesize that the age prediction task will not be easy for people over 65 years, because so few people probably can't represent the entire population of this age.

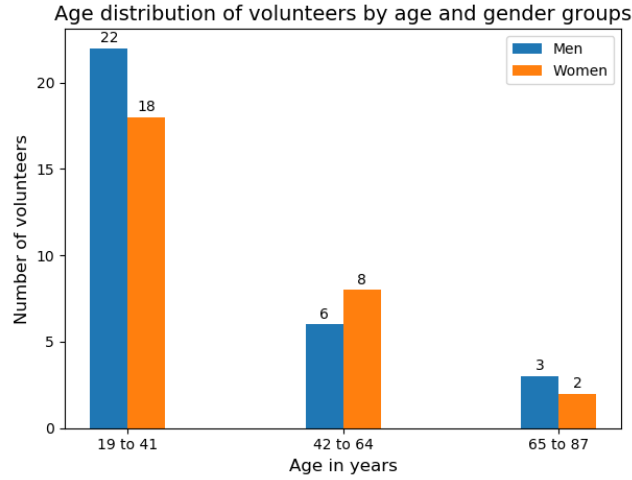


Figure 2.4: 3 unbalanced subgroups for the age prediction. Both men and women are present in each group.

Gender. 31 men and 28 women have taken part in the study. The distribution of the gender for each age group can be seen on Figure 2.4. Men and women are present in each group. We performed a Welch's t-test between the ages of men and women in order to see whether the age distribution is different. The obtained p-value (0.776) is greater than 0.05 and shows that the null hypothesis of identical average age cannot be rejected: the age distribution between men and women is not significantly different.

Dominant hand In total, 51 participants are right-handed, 5 are left-handed and 2 volunteers are ambidextrous.

Chapter 3

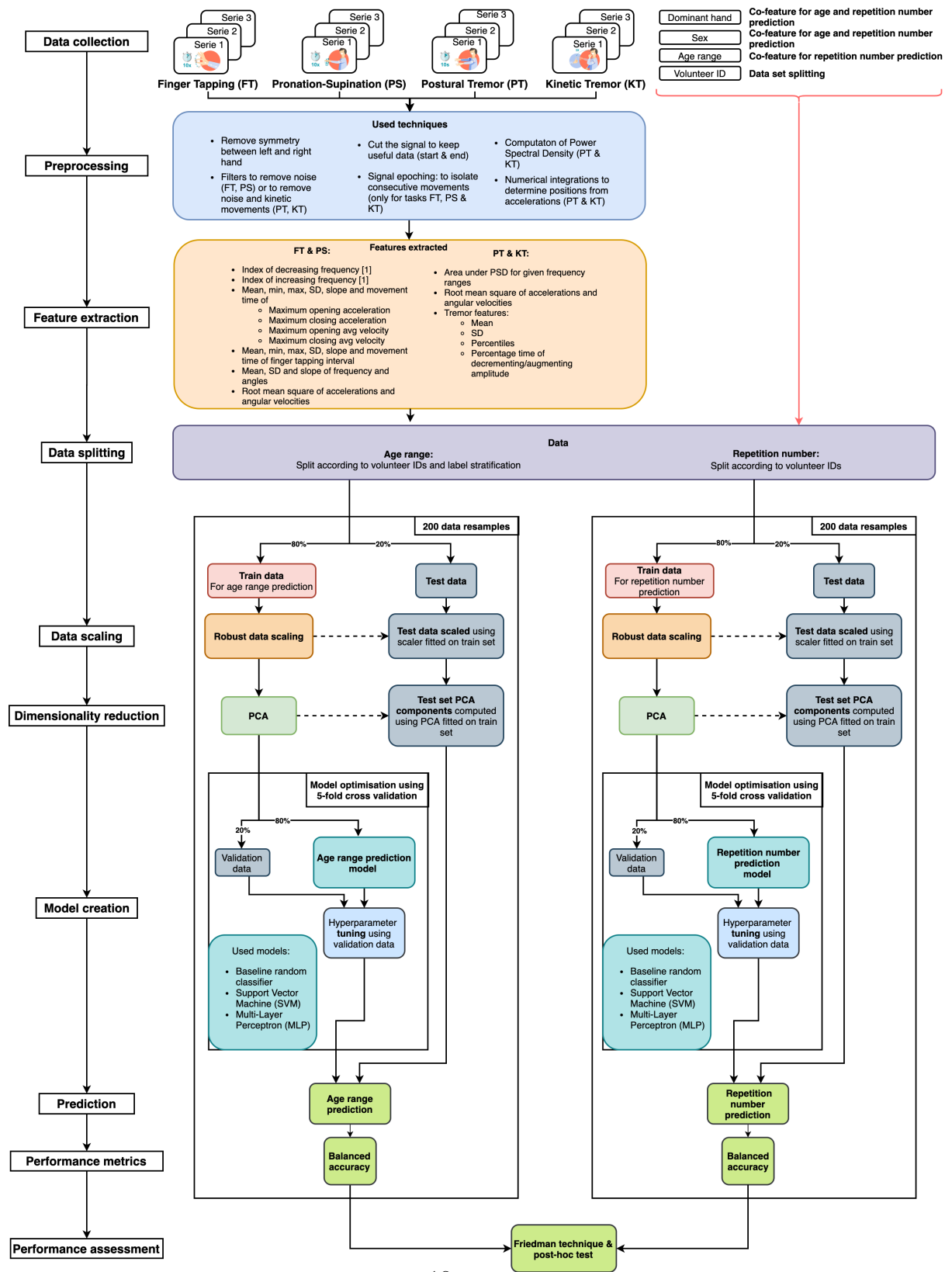
Solution overview

On the next page, a complete overview of the pipeline shows the steps to create and assess the performances of both prediction models.

First, the collected data is preprocessed using filters to remove the noise. Only the useful parts of the signal are kept and the hand symmetry is removed. This step is followed by feature extraction where, based on the task, a set of features are extracted such as opening and closing accelerations for FT and PS. For PT and KT, the features are computed with a focus on tremors, e.g. peaks and amplitudes of those tremors.

The data is then split into train and test sets based solely on the volunteers ID for the repetition number prediction (all the data of a volunteer is either in the train set or in the test set) or in addition to this on label stratification for the age range prediction. Before computing PCA, the data is scaled using a scaling technique robust to outliers. Once PCA has been fitted onto the train set, the train set and the test set are transformed using this PCA model.

At this point, the SVM and MLP models are optimised using 5-fold cross validation where the train set gets split further into a smaller train set and a validation set 5 times. Next, the models for the age range and the repetition number are trained on the complete train set and predict the labels of the test set. The predictions are compared to the truth values using the balanced accuracy metric. The steps from the moment the data set is split into train and test set represent one iteration. This iteration is repeated 200 times and both the individual scores and the means of the scores for each prediction are the final scores for each model. The means and the individual scores are then used to compute the ranks of the models and their difference is assessed using the Friedman technique. Finally, their performances are compared with two post-hoc tests.



Chapter 4

Preprocessing

Once the movements are recorded, the records need to be transferred from the tablets to a computer. Each file contains the data of one repetition of one task recorded on one hand. Thus, since there are four tasks and three repetitions of each task, and that the tasks are recorded on both hands, the collected data consists of 24 files per volunteer, where each file includes acceleration and angular velocities on the three axes and for each finger (thumb and index).

Tools4Patient has provided a Python package, *pysensormotor*, to read the records, convert the raw data into angles, visualize raw data and angles. Visualising the data has allowed to identify the preprocessing steps needed to later enable feature extraction. In this section, the preprocessing approaches common to each task are described. Then, for each task, the approaches specific to the task are presented.

4.1 Remove the hand symmetry

Since each task is recorded on both hands, the symmetry present on certain axes between the recordings of the left hand and those of the right hand were removed to be able to extract features in the same way on each hand. The datasheet[12] of the IMUs used by the SensorMotors uses Figure 4.1 and provides the orientations of accelerometers and gyroscopes. The IMUs are placed as such: if the right hand is placed flat on a table and the sensor is placed on the index finger, +Z goes up, +Y is in the extension of the finger and +X points towards the middle finger.

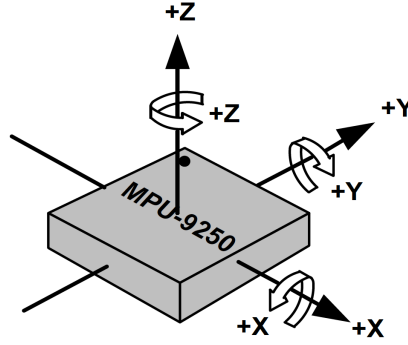


Figure 4.1: Axes orientation of accelerometers and gyroscopes

The axes on which a symmetry between the tasks produced by the left hand and those produced by the right hand can be observed are the following:

- the Y- and Z-axis for the angles and the angular velocities data
- the X-axis for the acceleration data

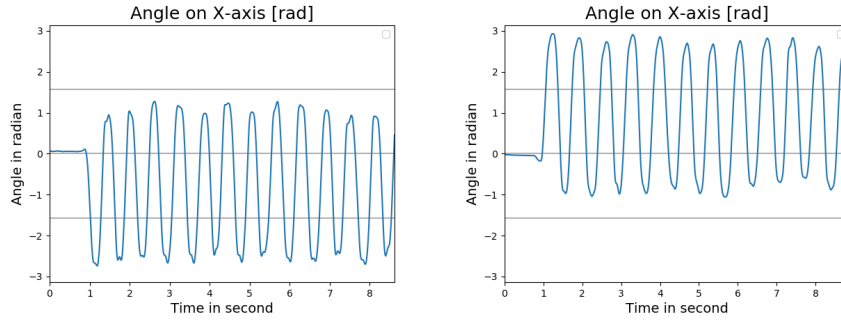


Figure 4.2: Angles recorded for the first repetition of the task pronation-supination of the hands. Left hand (to the left) and right hand (to the right) ¹

As an illustration, Figure 4.2 shows a plot of the angles according to time, computed along the X-axis for task 2, pronation-supination of the hands (PS), for the left and right hand. This plot highlights the symmetry between the left hand and the right hand for the Y- and Z-axis. To remove this symmetry, we have chosen to transform the signals of the left hand of the concerned axes in their opposite : each point of the signal is multiplied by -1. It is important to remember here that

¹Data from volunteer with ID 007, PS task, first repetition, left and right hands (records ST00701L and ST00701R).

for each record, the information about the hand from which it was recorded is still present in that record.

4.2 Finger tapping and pronation-supination

After removing the symmetry, some more steps are needed in order to prepare the records for feature extraction. As a reminder, the FT task is mainly a rotation around the X axis and during the PS movement, the main information can be found in the rotation around the Y axis.

4.2.1 Segment the signals

Since features need to be extracted from each individual movement in a record, it is important we choose a technique that allows to extract multiple time-windows from a signal. In signal processing, such a technique is called epoching where one time-window is called an epoch. We have chosen to use the epoching technique explained by Yokoe et al.[29] as a basis to split finger tapping as it allows to cut signals efficiently.

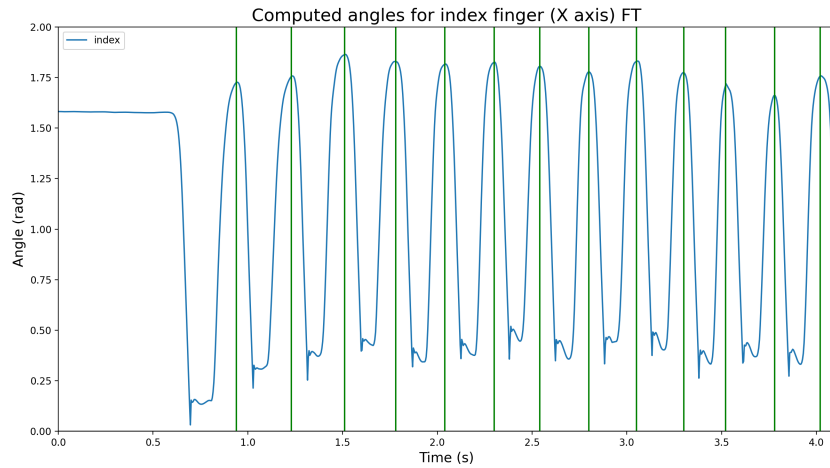


Figure 4.3: Green vertical lines represent the desired epoching for the finger tapping task on the index finger.²

²Data from volunteer with ID 001, FT task, first repetition, right hand (record ST00100R).

For finger tapping, volunteers are asked to start with their fingers in an open position. Thus, we have defined one finger tapping epoch as the time window between two maximum opening angles, where the maximum opening angle is the local highest value of the angle of the index. The expected result of our design choice can be found on Figure 4.3.

The same idea can be applied to the pronation-supination movement. Since the volunteer starts the movement with his/her palm pointing towards the ground, we have defined one pronation-supination epoch as the time window between two minimum closing angles.

Before being able to epoch both tasks, the start of the first movement must be determined in the records. This start time depends on the reactivity of each volunteer and is thus something that is not constant. Therefore, this start time has to be deduced directly from the data. Luckily, detecting this turns out to be manageable by an algorithm computing the difference between two data points. When a volunteer starts a finger tapping movement, the difference in the values on the X-axis of the angles computed for the index gives the amount of movement between two data points. In order to find the exact moment where these values start to differ suddenly, giving away the start of the movement, we have computed the difference between each point and the previous point.

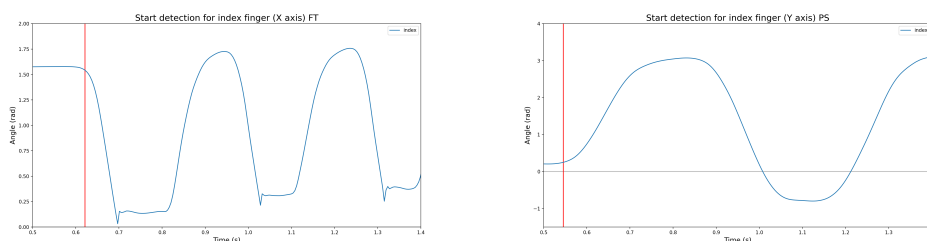


Figure 4.4: Red line indicates the start. Left : Start detection for FT. Right : Start detection for PS.³

Based on visual results, some experiments were made with a threshold value that has helped to define the start time of the first movement. According to us, setting the difference value 0.01 for this threshold has given the best visual results as it only removes a very insignificant part of the first epoch as can be seen on the left of graph Figure 4.4. Pronation-supination has the same sudden difference

³Data from volunteer with ID 001, FT task and PS task, first repetition, right hand (record ST00100R and ST00101R).

in the values in the beginning when a volunteer start to rotate his hand. Again, when experimenting with different threshold values the exact same value has given, according to our tests and for the same reason as for FT, the best results: 0.01. This can be seen on the right graph of Figure 4.4.

Once the start of the first epoch has been found, the cutting of the signals into multiple epochs can start. The same technique can be applied to both tasks so first it will be detailed for finger tapping followed by a description on how it applies to pronation-supination.

Looking back at Figure 4.3, multiple information can be used to detect when a movement starts and when it ends. The local maxima represent the moments when the index of the volunteer is at its peak, acting as an indicator for the end of an epoch and the start of a new one. The goal is now to search for local maxima by comparing them with neighboring values. We have chosen to do this by defining the minimum prominence of the sought-after peaks. The prominence measures the difference between the height of a peak and the lowest contour line that encloses no higher peak as shown on Figure 4.5.

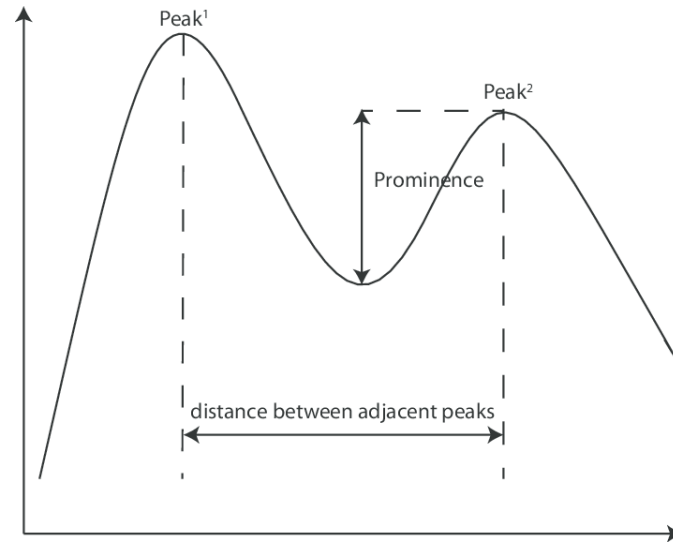


Figure 4.5: Prominence and horizontal distance between Peak 1 and Peak 2.⁴

⁴This figure is adapted from [15]

By analyzing the mean prominence of the peaks of all the records, all the peaks are captured by setting a minimum prominence of 0.5 radian. The result of the peak detection using the prominence can be seen on Figure 4.6 for FT. The advantage of prominence compared to a combination of minimum height and the number samples between each peak is that the former is far more flexible than the latter. The data set that is used here contains both records where peaks are very close to each other and peaks with a downward trend. Both of those cannot be perfectly detected using the combination mentioned before as the number of data points between each peak varies enormously and the use of a minimum height would fail to detect peaks for which the downtrend pushes them to go below this minimum height.

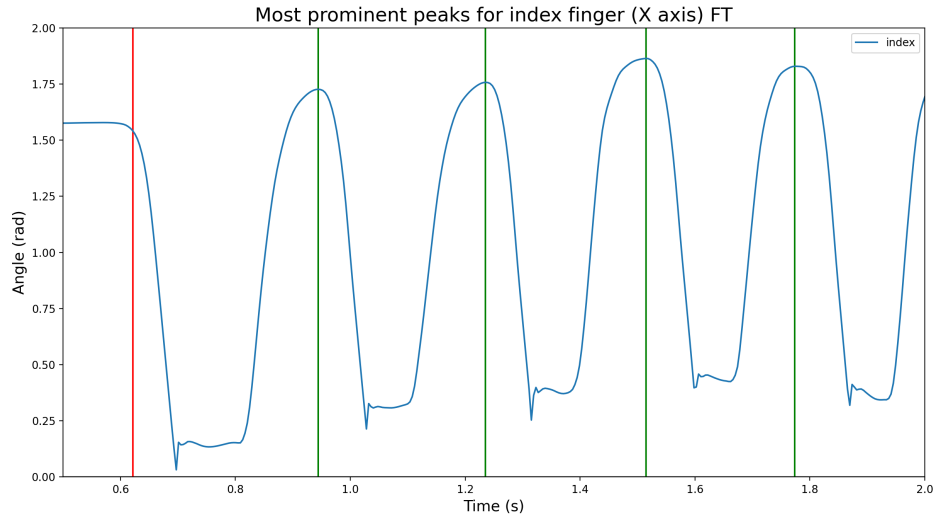


Figure 4.6: Red: detected start for finger tapping. Green: Most prominent peaks for index finger on the X axis for the FT task.⁵

The Y-axis from the PS task shows a pattern very similar to the one shown for FT. This has helped us to determine that peak detection would be a working technique. However, the peaks don't match with the end of a movement but rather with the maximum opening angle which occurs in the middle of an epoch. A solution to this is to invert the signal, the minima become maxima and vice versa. Searching for the most prominent peaks on the inverted data, gives the the peaks at what was previously the minima. The result of the peak detection using the prominence can be seen on Figure 4.7 for PS.

⁵Data from volunteer with ID 001, FT task, first repetition, right hand (record ST00100R).

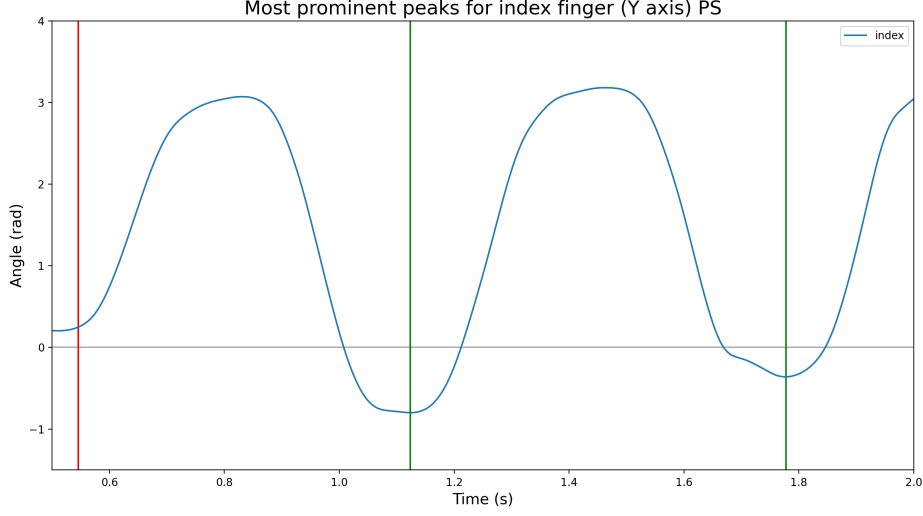


Figure 4.7: Red: detected start for pronation-supination. Most prominent peaks for the index finger on the X axis for the PS task.⁶

The data for both tasks being cut in epochs, there is still an important factor to take care of. On closer inspection of the first recorded point by the accelerometers for each axis, it can be noticed that they don't exactly start at 0g neither for the index nor the thumb. There is a plausible explanation for this offset. When the hands of a volunteer are at rest waiting for the exercise to start, there is a force applied to the accelerometers: gravity. Depending on the position of the hand, the force applied by gravity will differ for each axis. According to us, leaving this bias in the data would be a problem so we searched for techniques that remove gravity.

For the FT task, since only the fingers are moving and the rotation of the hand is almost insignificant, the gravity component doesn't vary a lot. Therefore, the gravity component can be removed by centering the data on each axis using the mean of the first data points : the mean of the first data points give the value for the gravity vector applying on an axis while the hand is stationary. This way, we know that if we subtract this value from our data, we have removed the gravity component.

The last step is to keep only a fixed amount of epochs for future feature extraction. This amount is chosen during the recording sessions as we have aligned our instructions with the instructions given by MDS-UPDRS. For FT and PS, only the first 10 epochs are kept. Keeping more epochs would be irrelevant as physicians following the MDS-UPDRS rate a patient on those first 10 epochs.

⁶Data from volunteer with ID 001, PS task, first repetition, right hand (record ST00101R).

4.3 Postural and kinetic tremors tasks

For the PT and KT tasks, future feature extraction techniques focus on features related to tremors. For this purpose and according to what can be learned from the state of the art, we have decided to compute the power spectral density and the displacement of the fingers due to tremors from the acceleration data. The first preprocessing step, already described above, consists in removing the symmetry between the left and right hands. The other steps are described thereafter.

4.3.1 Segment the signals

Once the symmetry between the left and the right hand has been removed, the signals need to be segmented (accelerations, angular velocities and angles for the three axes) to keep only the useful parts.

For the third task, postural tremor of the hands (PT), we cut and remove the signals before the first half second since the recording begins a little bit before the beep sound of the tablet and we cut and removed the signal after 10,5 seconds to the same recording length for each participant : 10 seconds.

For the fourth task, kinetic tremor of the hands (KT), we segment the signal to keep only the first three "nose-finger" movements. In order to do that, the start and the end of each movement are detected and the parts of the signal before the first movement and after the third movement are cut and removed. The end of a movement is considered as the start of the next movement.

Start detection of the KT task More precisely, we have defined the start of the movement as the moment where the index finger begins to move towards the nose. This rotation is always clearly visible on the Z-axis of rotation of the angles so we look at this axis to detect the beginning of the movement. On this axis, the difference of each two consecutive points is computed, a threshold of 0.004 is fixed and when five consecutive points of this difference reach this threshold, we consider that the movement has started. A visual inspection has allowed to find this threshold and to check that the start was detected correctly on each record of each volunteer.

End detection of the KT task Once the start is found, it is also interesting to automatically detect the end of the movement, which is also the beginning of the next movement. To detect this end, the two next peaks on the angles are found, representing the two next moments when the finger touches the nose.

Taking the point between these two points enables to detect the next start after this point with the same process as for finding the start of the first movement. This new start represents the start of the next movement and thus also the end of the current movement.

As an illustration, Figure 4.8 shows the angles on the Z-axis according to time for the KT task. The red bar shows the start of the first movement, the green bars delimit the movements and the blue bar represents the end of the third movement, and thus the end of the signal kept.

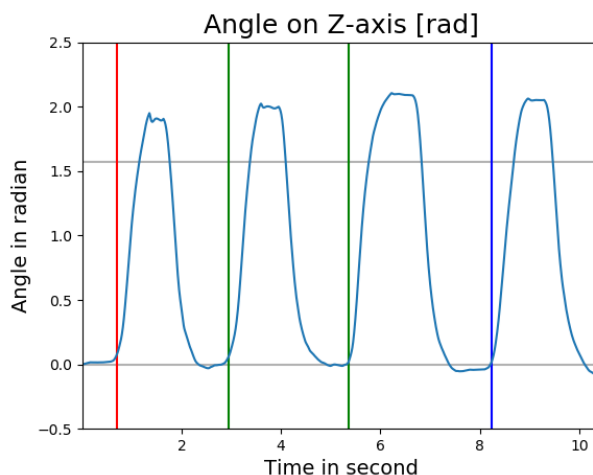


Figure 4.8: Segmentation of the movements for the KT task. Red: start of the first movement. Green : delimitation of following movements. Blue : end of the third movement.⁷

4.3.2 Preprocessing acceleration data

In order to obtain displacements of the fingers due to tremors from the acceleration data, the acceleration components due to kinetic movements and gravity first need to be removed. This is done by applying a band-pass filter to each axis of the acceleration data.

Postural tremors. We used a band-pass filter (*scipy.signal.butter*, order 6) with the frequency range [3-60] Hz. In this way, the gravitational component of the acceleration is removed and eventually some small oscillations of the fingers

⁷Data from volunteer with ID 001, KT task, first repetition, right hand (record ST00103R).

(frequencies below 3 Hz) too, as well as some noise (frequencies above 60 Hz). Indeed, we assumed that the tremor frequencies will never be above 60 Hz, and a visual inspection of the power spectral density has confirmed this hypothesis.

Kinetic tremors. We used a band-pass filter (*scipy.signal.butter*, order 6) with the frequency range [3-60] Hz. As for the PT task, we assume that the tremors will not have frequencies below 3 Hz and above 60 Hz, and that movement frequencies are not above 3 Hz.

Note that before applying the band-pass filter, the acceleration data must first be centered to 0. Indeed, due to the gravitational component of the acceleration, the acceleration of certain axes are not centered to 0, and if they are not, a side effect appears at the extremities of the filtered signal. In order to center the data to 0, we compute for both tasks the average of the first twenty data points (the hand is then motionless) and we subtract this average from each point of the signal.

4.3.3 Double integration of acceleration data

Once the acceleration component assumed to be due to tremors is found, we integrate (*scipy.integrate.cumtrapz*) it twice. But since a drift due to the sensors is still present, we use the function *signal.detrend* to cancel the linear trend caused by the drift between each integration.

4.3.4 Power Spectral Density

As seen in the state of the art section, the power spectral density can be useful for studying tremors. There are several techniques to produce estimates of the power spectral density. Here, we have decided to use the Welch's method (*scipy.signal.welch*). The latter is an improved version of the periodogram. The periodogram uses the square of the module of the Fourier Transform divided by the number of points constituting the signal to provide an estimate of the power spectral density. However, this technique is not a consistent estimator of the power spectral density : there is a bias for signals of limited length and a high variance of the spectral estimate for a given frequency that does not decrease when the length of the signal increases.

The Welch's method[4, 28] is one of the existing techniques improving the periodogram. This method is asymptotically unbiased and proposes the following principle : the signal $s[n]$ of length M is segmented into K segments of length L .

The K segments may partially overlap each other. A window is applied on each segment to reduce bias and the periodogram of each modified segment is computed. The result of the Welch's method is the average of these periodograms, which allows to reduce the variance. This method is therefore defined as followed :

$$S_W(\omega) = \frac{1}{KLU} \sum_{i=0}^{K-1} \left| \left(\sum_{n=0}^{L-1} w(n)s[n+iD] \exp(-j\omega n) \right) \right|^2 \quad (4.1)$$

Where :

- s is the signal
- D is the number of points between each start of segment (each segment is offset by D points)
- L is the length of the segment
- K is the number of segments
- ω is the angular frequency
- $w(n)$ is a general window of length L ,
- $U = \frac{1}{L} \sum_{n=0}^{L-1} |w(n)|^2$ is a constant introduced so that each periodogram is asymptotically unbiased.

To use Welch's method, a number of parameters must be chosen : the window, the percentage of overlapping between the segments, and the length of each segment that will induce, for a given overlap, the number of segments.

Let us now discuss the way these parameters were chosen. Firstly, the documentation of the scipy function[5] informs us that "*the default Hann window with an overlap of 50% is a reasonable trade off between accurately estimating the signal power, while not over counting any of the data.*". Since Welch suggested[28] as a reasonable choice a window close to the cosine arch spectral window and that the Hann window is such a window, these default Hann window provided by scipy was used, with an overlap of 50%. Then, the length of the segments was chosen. It induces, as said above, the number of segments used to make the average. Here, the trade-off is the following : the wider the window, the higher the frequency resolution, but the fewer the segments and the less precise the estimate. In order to have an idea of plausible values, the documentation of the equivalent matlab function, *pwelch()* has provided some inspiration. Here, by default, the segments are as long as possible "to obtain as close to but not exceed 8 segments with 50% overlap"[27] with an Hamming window, which is quite similar to the Hann's window. Taking this into account, the following length for the segments were chosen :

- For the PT task, since 10 seconds of the records are kept and that the sampling frequency is around 250 Hz, this makes a total of roughly 2500 samples. We have chosen to use segments of length 512 to have, with an overlap of 50%, about $\left\lfloor \frac{2*2500}{512} \right\rfloor - 1 = 8$ segments. This results in a frequency resolution of about 0.5 Hz.
- For the KT task, 3 "finger-nose" movements are kept. Since the duration of one movement is not fixed, we have computed the average value of all "finger-nose" movements of all patients. This average is 1.7 seconds. We have decided to choose segments of 256 points to have, with an overlap of 50%, more or less $\left\lfloor \frac{2*3*1.7*250}{256} \right\rfloor - 1 = 8$ segments, according to the duration of the three movements of each records. This results in a frequency resolution of about 1 Hz.

Chapter 5

Feature Extraction

Specific features are extracted from each record of the preprocessed data. In this section, this process is described according to the type of movement.

5.1 Features based on movement

The literature on FT movement characterization with devices provides some interesting features to extract[29, 23] which is discussed in subsection 1.3.2 . Task-based information, such as knowing how an epoch matches with a movement, is very helpful during this phase. Since a lot of similarities exist between the FT and PS tasks, the feature extraction of both tasks have been handled together. For this reason, the extracted features are described in general for FT and a description on how it applies for PS is given.

To understand the elements that are being extracted from the data, a small introduction to the different part of a movement is required.

For FT, we cut one movement into multiple sub-parts: the closing phase, the contact phase and the opening phase. These sub-parts of a movement are easier to work with when dealing with features that only depend on one sub-part such as the opening velocity features. Working with smaller parts also means working with less data and speeds up algorithms that perform multiple comparisons between the data points of one record.

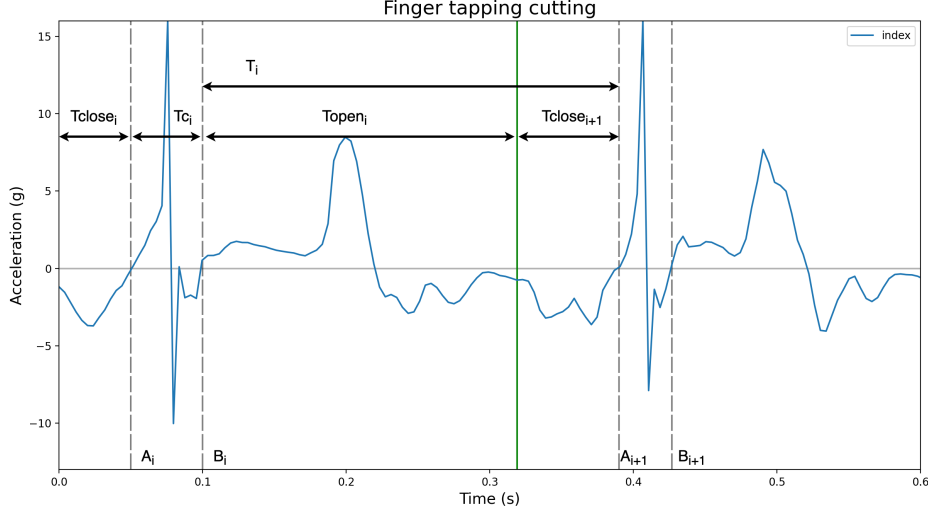


Figure 5.1: Z-axis acceleration data for FT: movement segmentation¹

On Figure 5.1, A_i and B_i indicate respectively the time where the fingers make contact and when the fingers separate where i is the number of finger taps. T_i indicates the time between two finger tapping contacts. Tc_i is defined as the finger contact interval. The open and close intervals for one single finger tapping movement are defined as $Tclose_i$ and $Topen_i$. $Tclose_{i+1}$ is the start of a new finger tapping movement where the volunteer closes its fingers after reaching the peak opening angle.

5.1.1 Accelerometer-specific features

As a reminder, for FT, volunteers are asked to hold their hand straight in front of them. Thus the main features of FT recorded by the accelerometer are extracted from the Z axis. The index stays on this axis when going up and down.

Using the cutting described previously, two components are of interest: the closing phase and the opening phase. For the closing phase, we are interested by knowing what the **maximum closing acceleration** is as well as how much time it took the participant to get to that maximum acceleration point, i.e. **movement time for maximum closing acceleration**. Looking at Figure 5.1, the absolute maximum closing acceleration is reached during $Tclose_i$ before Tc_i . The start of this contact interval is characterized by a big peak representing the shock of

¹Data from volunteer with ID 001, FT task, first repetition, right hand (record ST00100R).

the contact between the two fingers. The point before the peak can be found by analysing the slope of the data until coming upon a point where the slope exceeds a threshold giving the moment before the peak of contact occurs. The features from the closing phase can now be retrieved more readily if the data is limited to the points between the start and the moment before the point of contact, in other words the T_{close_i} phase. During this phase, the smallest value, which represents the absolute maximum closing acceleration can be identified. This acceleration is negative for the index and the reason behind this is because during the closing phase, the index goes in the opposite direction of the positive Z axis, thus going negative (see section 4.1 for the different axis directions). Finding this maximum value also gives the total time needed to reach this point from the start of the movement. This movement time has been discussed by Yokoe et al.[29] and discloses whether the volunteer did the movement at a slower or higher pace.

Analysing the data after T_{c_i} shows that the time of contact differs between the two epochs. This can indicate that the volunteer is accelerating and is discussed later when talking about non-specific features. The maximum value of the acceleration between the end of the contact phase and the end of the epoch, T_{open_i} , is the **maximum opening acceleration**. Computing when this maximum value occurs gives the **movement time for the maximum opening acceleration**.

Each of these values is extracted for one epoch. To get an overview of the complete movement of each task, we compute the mean, the standard deviation and the slope for each of the 10 values gathered for the maximum opening and closing acceleration as well as the movement time for the maximum opening and closing acceleration. These different computations are based on the description of the extracted features given by Stamatakis et al.[23].

5.1.2 Gyroscope-specific features

In the gyroscope data, the informative axis for the FT movement is the X-axis.

As for the accelerometer, we divide the signal of one epoch into multiple sub-parts. For the segmentation of the FT movement, intervals T_i , T_{close_i} , T_{open_i} were defined. These intervals also apply to the PS task with two main difference compared to FT. The first difference is that the data is no longer acceleration data but gyroscope data. The second difference is that there is now a maximum rotation interval, T_{max_i} , instead of a finger contact interval. This can be seen on Figure 5.2.

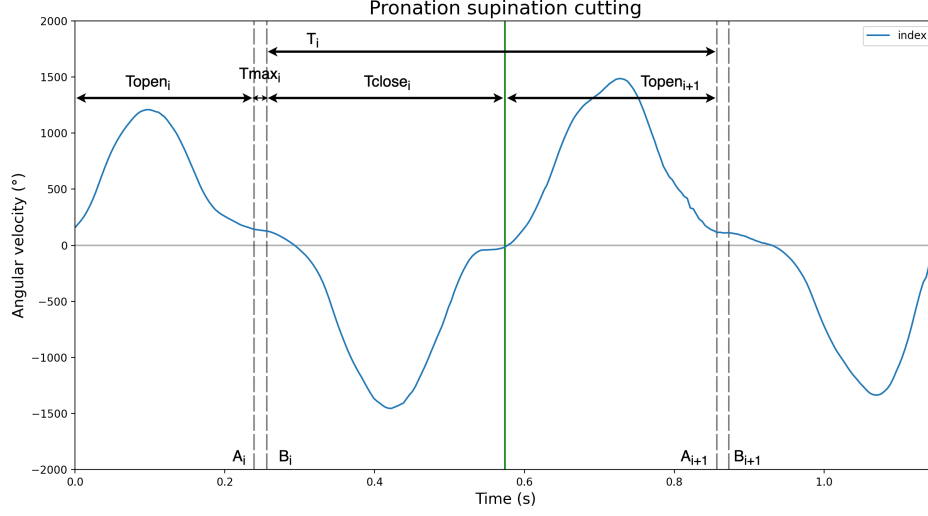


Figure 5.2: X-axis acceleration data for PS movement of the hands: movement segmentation²

Using the same logic as used during the extraction of the accelerometer-specific features, we have extracted from the gyroscope data the maximum opening and closing angular velocity. The movement time for maximum opening and closing angular velocity is also extracted using the same technique mentioned in the accelerometer section.

5.1.3 Non-specific features

Some of the extracted features don't belong to either accelerometer features or gyroscope features. These features give more information about how the volunteer performed in the task.

With the angles, features relative to movement amplitude can be computed. Extracting the maximum amplitude of each epoch allows to compute two elements: the maximum amplitude slope and the decrementing or augmenting index of amplitude. The former serves as a backup to indicate whether the volunteer manages to reach the same amplitude as when he/she started the task. The latter is a measure computed with a statistical T-test. In this T-test, the mean values of two samples are compared: the amplitude of epochs up to the one currently being analysed and the amplitude of the epochs that still have to be performed. This test allows to

²Data from volunteer with ID 001, FT task, first repetition, right hand (record ST00100R).

determine the epoch at which the amplitude of the finger of the participant starts to decrement or augment significantly. Knowing when this happens, allows to see which part of the volunteers speed up or slow down at a certain time during the task[23].

As for movement amplitude, the slope of the movement frequency and the decrementing or augmenting index of movement frequency is computed. Movement frequency is defined as the inverse of the time needed for the participant to perform one epoch.

Moreover, halts are an important factor to score the evolution of Parkinson Disease. To study whether this parameter would fluctuate given a volunteer’s age, we included it in the features.

We included the Root Mean Square of each axis of the acceleration data and the angular velocity data. Here the root mean square is defined as the square root of the arithmetic mean of squares of the signal s , i.e :

$$RMS_{finger,axis,data} = \sqrt{\frac{1}{N} \sum_{i=0}^{N-1} s[i]_{finger,axis,data}^2} \quad (5.1)$$

As a last set of features, we included the actual speed of the participant for both tasks. The number of finger taps per second and the number of hand turns for the duration of the 10 epochs of each task was added to the features.

5.2 Features based on tremors

For the two last tasks, PT and KT, the main goal is to extract relevant information characterizing the tremors. For this purpose, we have chosen to use the power spectral density and the positions computed in the preprocessing part as well as the accelerations and angular velocities to extract different features. This subsection describes which features are extracted for which signals, and how. The same features are extracted in the same way for the two tasks, so this process is described simultaneously for the two tasks.

5.2.1 Features extracted from accelerations and angular velocities

From accelerations and angular velocities, the root mean square of these signals have been extracted. Indeed, it is expected that records with a lot of tremors

will have a greater root mean square. In addition to that, in order to not take into account the accelerations and angular velocities due to kinetic movements, the root mean square has been computed after having centered them (only for accelerations), and filtered them as explained in the preprocessing section.

Concretely, for acceleration and the angular velocity data, the root mean square of each finger and each axis is computed with the same formula as mentioned in the previous section.

5.2.2 Features extracted from positions

As a reminder, the positions are integrated from the accelerations after having centered and filtered them to remove kinetic movements, gravity and some noise. In this way, only positions and variations of positions due to tremors are obtained. The goal is to be able to extract features characterizing the tremors more easily. More particularly, we have extracted information characterizing the amplitude of the tremors and the moments at which they occur. An example can be found on Figure 5.3.

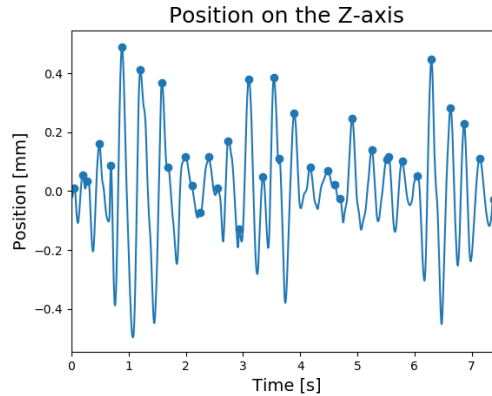


Figure 5.3: Position according to time for a repetition of the KT task. Peaks are represented by the blue points.³

Firstly, for each signal representing the positions according to time for a given axis (X, Y or Z), the same process has been applied: all the peaks are found and for each peak, its prominence is analysed (i.e what is consider as the amplitude of the tremor) and the time at which it occurs. Then, on the one hand, the mean, the standard deviation and several percentiles : the 5th, 25th, 50th, 75th, and

³Data from volunteer with ID 001, KT task, first repetition, right hand (record ST00103R).

95th are computed. On the other hand, the percentage of time at which the amplitude of the tremors increases significantly and the percentage of time at which it decreases significantly is computed. To assess that the amplitude has increased (resp. decreased) significantly, we used a t-test : for each tremor, the mean amplitude of previous tremors is compared to the mean amplitude of this tremor and the following. If the mean of the previous tremors is significantly lower (resp. higher) for this t-test and that of the following tremor ($p\text{-value} < 0.05$), the moment of the first of the two tremors is considered as the time where there is a significant increase (resp. decrease) of amplitude. If there is no significant increase (decrease) in amplitude, the moment of the end of the task is saved. It is worth noting that to be able to compare the times found between each participant, and since each participant did not take the same time to accomplish the last task (KT), we decided to normalize this time between 0 (start of the task) and 1 (its end) by dividing the time found by the total duration of the task. In this way, the value of the feature is more relevant for the KT task and the normalization has no impact for the PT task, which takes 10 seconds for each participant.

5.2.3 Features extracted from power spectral density

To be able to extract informative features from the power spectral density (PSD) already calculated, analysing the curves represents the first step.

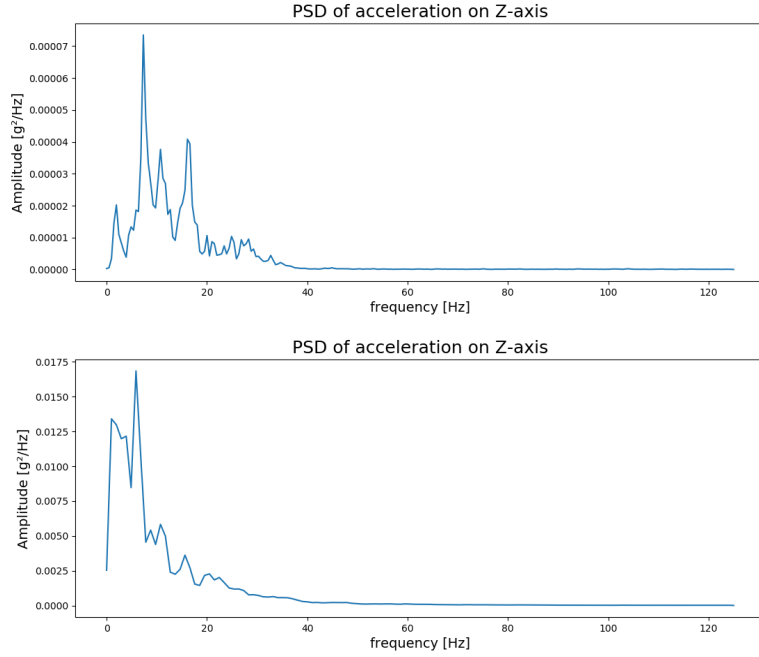


Figure 5.4: Top : PSD for the PT task. Bottom : PSD for the KT task.⁴

As an illustration, the Figure 5.4 represents the PSD of acceleration data on Z-axis, for one repetition of a volunteer for the PT task (top), and for one repetition of the KT task (bottom). The first peak for the KT task has a much higher amplitude than the PT task. According to the nature of both tasks, this peak is due to the kinetic movements much more present for the KT task because of the "finger-nose" movements, than for the PT task where there should only be possible slight oscillations of the fingers.

Since several peaks may be present or not at different frequencies according to the volunteers, we have decided to keep as features the area under the curve for several frequency ranges, one feature for one range. The frequency ranges are the following : 2-5 Hz, 5-10 Hz, 10-15 Hz, ..., 55-60 Hz. Indeed, a manual inspection has showed, according to participants, several peaks higher than power of the presumed "noise", present until 50 Hz. To be sure, we have kept the area until 60 Hz, taking into account that the features used to build the models will then be selected among all the features, and therefore it is better to extract too many features than not enough.

⁴Data from volunteer with ID 020, PT and KT tasks, first repetition, right hand (records ST02002R and ST02003R).

Chapter 6

Data visualization and dimensionality reduction

The features extracted from the records now compose a data set in addition to the volunteer's data. Building some intuition about relevance and redundancy of these features is the first step of this chapter. The relevance and redundancy of the features is examined by using the Pearson's correlation coefficient. The high correlation between some features leads us to choose the PCA transformation as a dimensionality reduction technique. How this technique can be useful is briefly described. The two first PCA components are then used to better visualize the data. Finally, the interpretation of what has been found in this chapter leads to a proposition to manipulate the data set to build another one with potentially denoised samples.

6.1 Relevance and redundancy of the features

Knowing to what extent the extracted features are relevant in regard to the labels to predicts, and redundant with each other is the first step of the feature analysis here. For this purpose, the Pearson's correlation coefficients between each feature and the outputs (repetition number, age) have been computed to get some intuition about the relevance of the features, and between each pair of features to get some intuition about their redundancy. The Pearson's correlation coefficient ρ of two random variable X, Y is given by the following equation :

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y} \quad (6.1)$$

Where cov is the covariance, σ_X (resp. σ_Y) is the standard deviation of X (resp. Y), E is the expectation and μ_X (resp. μ_Y) is the mean of X (resp. Y). The Pearson's correlation coefficient for a sample of data can be expressed by replacing covariance and standard deviation by the estimates of the covariance and the standard deviation: for n pairs of data $(x_1, y_1), \dots, (x_n, y_n)$, the sample Pearson's correlation coefficient r_{xy} is :

$$r_{xy} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}} \quad (6.2)$$

Where $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$ and $\bar{y} = \frac{1}{n} \sum_{i=1}^n y_i$ are the two sample means. The correlation coefficient between two features is a real number belonging to the interval $[-1, 1]$. A correlation coefficient equal to 0 means that there is no linear dependency between the two features (but another non-linear dependency can exist), and a value of 1 (resp. -1) can be interpreted as the existence of a perfect positive (resp. negative) linear correlation : all points lie on a line for which the first feature increases (resp. decreases) when the second feature increases. Note that since only the existence of a linear dependency between two features is inspected, and not their sign (positive or negative), the absolute value of the correlation coefficients has been computed.

6.1.1 Relevance

The 30 features with the highest correlation coefficient for each output label are shown in Figure 6.1. The correlation between the features and the repetition number seems very low (greatest correlation coefficient: 0.187), which suggests either that the extracted features are robust to the training effect and that the prediction of the repetition number is hard, or that the features are influenced by the training effect, but this relation is not linear. The correlation coefficients w.r.t the age are higher, but there is no strong linear trend, which suggests that a linear model will not be appropriate.

Moreover, among the 30 highest-correlated features, 25 come from KT task, 3 from PT task, and 2 from FT tasks for the repetition numbers prediction. For the age prediction, 20 features come from the PT task, 8 from the KT task and 2 from the FT task. Thus, the most correlated features are extracted from tasks related to tremors (KT, PT tasks) : when considering linear relations between the features and the age or the repetition number, tasks related to tremors seem to be the most informative. However, both for the age and the repetition number, these correlation coefficients remain low and maybe non-linear tools could highlight other informative features, based on tasks related to tremors or not.

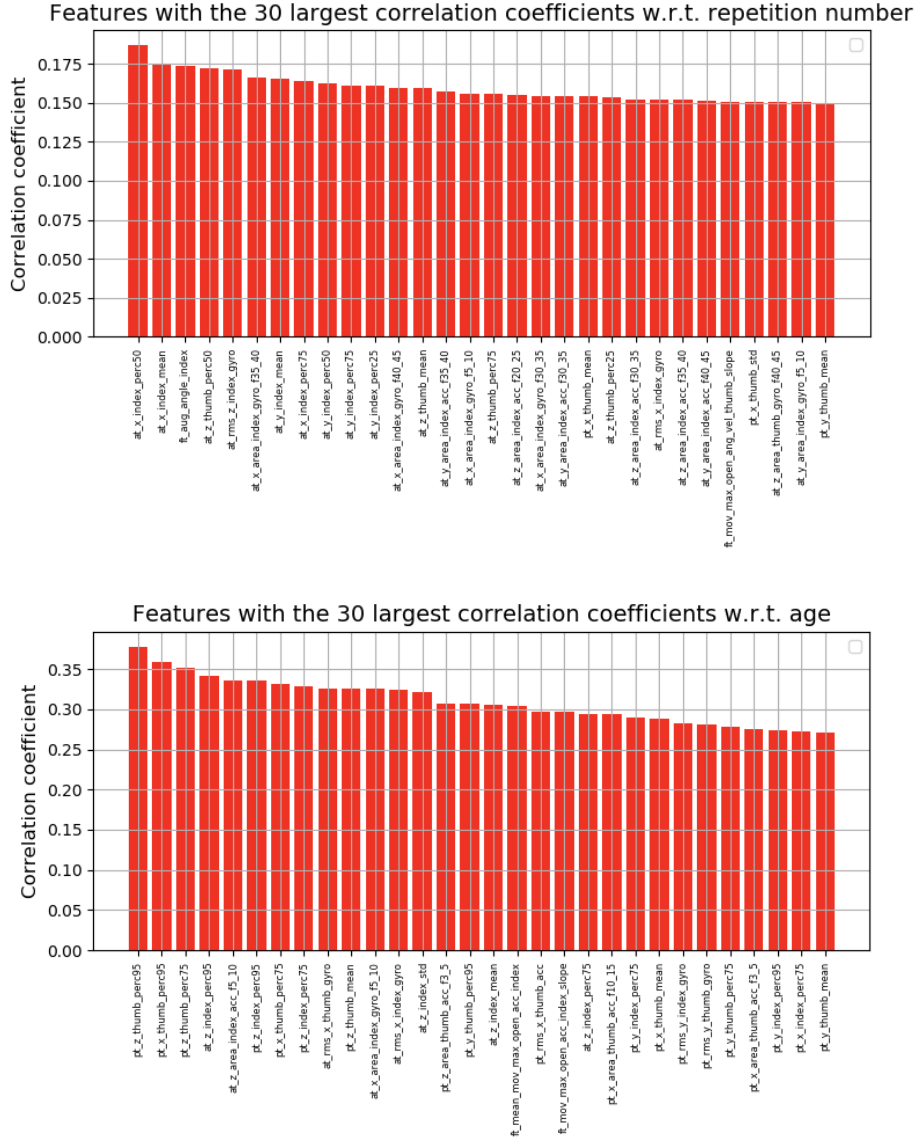


Figure 6.1: 30 features with the highest correlation coefficient w.r.t the repetition number (top) and the age (bottom).

6.1.2 Redundancy

Computing sample correlation coefficient between each pair of features gives the correlation matrix $\mathbf{R}$ where each element $r_{i,j}$ is the sample correlation coefficient between features i and j .

This matrix is shown as a heatmap in Figure 6.2 : all values of correlation coefficient are represented by a different intensity of the color red. The features are numbered in the following order :

- Patient's data : from 0 to 5
- Features extracted from FT task : from 6 to 60
- Features extracted from PS task : from 61 to 150
- Features extracted from PT task : from 151 to 360
- Features extracted from KT task : from 361 to 570

Knowing that, the three clusters present in the heatmap can be better understood: the features of the same task seem highly correlated with each other. The blue, yellow, green and purple squares represent the correlations of features extracted from FT, PS, PT and KT tasks respectively. Moreover, features from FT task also seem to be correlated with features from PS task (pink squares). This heatmap suggests that a lot of the features are redundant and that a tool like a PCA transformation could be useful to remove this redundancy and express the features in a lower dimensional space.

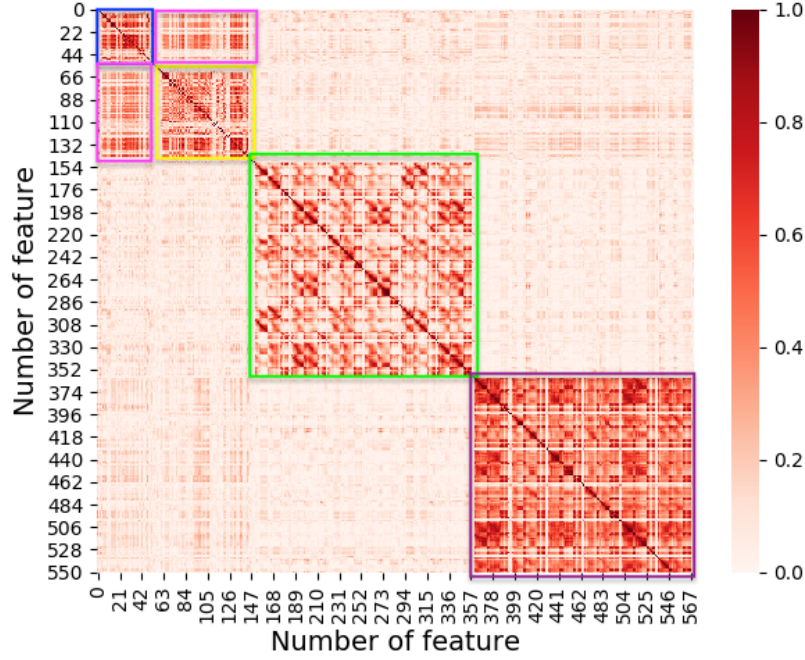


Figure 6.2: Heatmap of the correlation coefficient matrix. The blue, yellow, green and purple squares represent the correlations of features extracted from FT, PS, PT and KT tasks respectively. Pink squares represent the correlations between FT and PS tasks.

6.2 Principal Component Analysis and dimensionality reduction

The feature extraction gives a high dimensional data set of 570 features. Then, subsection 6.1.2 has shown that some features are highly correlated with each other. We are thus looking for a technique that could decorrelate the features while expressing them in a lower dimensionality space. A well-known dimensionality reduction technique that enables this is Principal Component Analysis. This technique uses the eigenvalue decomposition and aims to find a linear transformation of the features to express them in a new coordinate system where all the new variables are linearly decorrelated. The first coordinate, called first principal component, is the one with the greatest variance. The second coordinate has the second greatest variance, and so forth. PCA is often used as a dimensionality reduction technique : the number of components to be kept can be determined in order to preserve a

given percentage of the variance in the original data.

Moreover, it is worth noting that before applying the PCA transformation, the features are standardized according to the z-score : each value x_i of a feature sample $\{x_1, \dots, x_n\}$ is replaced by its z-score as shown in Equation 6.3.

$$z_i = \frac{x_i - \bar{x}}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2}} \quad (6.3)$$

where $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$ is the sample mean.

Benefits of PCA

Firstly, PCA reduces the model computation. Time and memory are two important factors when training models. Building a model using too many features may cause very high memory usages as well as very long building times. The PCA transformation allows to reduce the dimension of the data, and reducing this may solve both problems.

Then, PCA and other dimensionality reduction techniques allow to avoid the curse of dimensionality problem: as the dimensionality increases, so does the volume of the space in which the training examples exist, meaning that the available data may become sparse. The amount of data that would then be needed to perform tests with statistical significance could grow exponentially. In this study, where the amount of volunteers is limited, this would be a serious problem.

Drawbacks of PCA

A first drawback is the lack of interpretability of the new variables. If the features in the original space are meaningful, this is no longer the case for the new variables computed by the PCA transformation. That could be inconvenient as it removes the ability to analyse which features influence what needs to be predicted, i.e. the age and the repetition number. However, the Pearson's correlation coefficient gives information about linear dependency between the features and the age and repetition number, and other tools to analyse non linear dependency exist and could be used before applying PCA if more information about the relevance of the features is needed.

The second disadvantage is that PCA is not a feature selection technique. Even if the new feature space has a lower dimension, not all features are meaningful with respect to the target values. Noise can still be present in the new variables.

6.3 Visualization of the two first principal components

The high dimensionality of the features makes it difficult to visualize them. After the PCA transformation, data can be more easily interpreted by examining the two first principal components : the two new features that have the greatest variance, thus that express the most variance that is present in the original data. The two first principal components cumulate 29.2 % of the total variance of the initial features. The first principal component according to the second is shown on Figure 6.3 with colors representing repetition number (left) and age (right). On these graphs, repetitions and ages seem to be randomly distributed and no trend (linear or another) that could give an intuition of an appropriate model can be distinguished. However, many outliers are present in the graphs and the distribution of the ages and repetitions numbers cannot be easily seen for points which are not outliers.

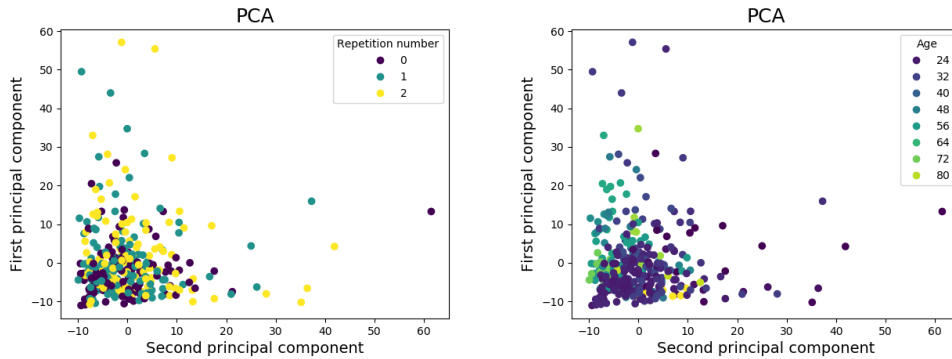


Figure 6.3: First principal component according to the second with colors representing repetition number (left) and age (right).

The principal component plots without the outliers are visible on Figure 6.4. In these graphs, all data samples where at least one of the two principal component values is not between the 5th and 95th percentiles have been removed. The left graph (with colors representing repetition numbers) does not give more information and on the right graph (with colors representing ages), no real cluster or trend appears either. Several reasons could explain this lack of trend: either there is really no trend or cluster for the ages and repetitions of the data, or there are but the PCA can't highlight them. Indeed, only 29.2 % of the variance of the initial data is kept in the first two components. This percentage may not be high enough to reveal a trend, especially since, as shown in Figure 6.3, there are many outliers that are probably responsible for a significant part of the initial variance.

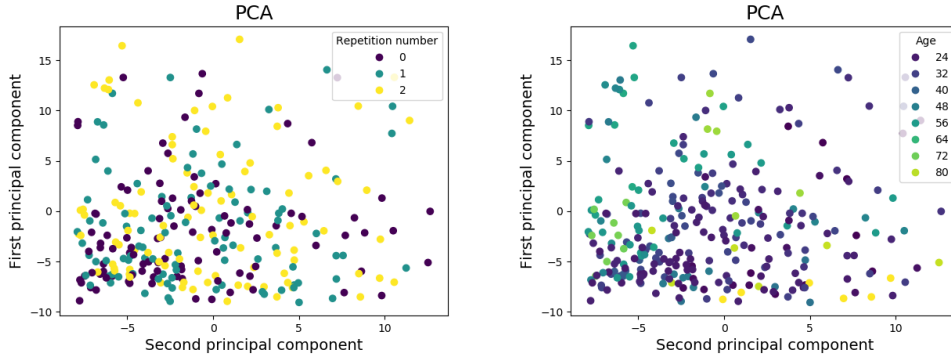


Figure 6.4: First principal component according to the second with colors representing repetition number (left) and age (right). In these graphs, outliers (samples where the value of at least one of the two principal components is not between the 5th and 95th percentiles) have been removed.

6.4 Reduce the noise for the age prediction task

The correlation coefficients of the features according to the repetition number and the PCA plots allow to hypothesize that the training effect is not very marked. Moreover, the difference in the values of a feature for the different repetitions of a task for the same volunteer is thus expected to be considered as a noise effect. To assess this hypothesis, we have created a second data set based on the first one, but where the six samples from a given volunteer are summarized in one : for each feature, we have kept the median values of the six ones. This new data set is thus composed of the same number of features as the previous one, but with only 59 samples of potentially "denoised data" instead of 354, since there are 59 participants in this study.

In order to get some intuition about how the use of denoised data can improve or not the predictions, a similar analysis about relevance, redundancy and PCA has been carried out. The 30 largest correlation coefficients between each feature and the age for the denoised data are shown on Figure 6.5. While they are between 0.27 and 0.38 for the normal data, the correlation coefficients are now between 0.36 and 0.52. Among the 30 features, 19 are in common with the 30 features with the largest correlation coefficients for the age of the normal data. And among these 19 features in common, 18 features have an higher correlation coefficient on the denoised data than in the normal data, which suggest that taking the median of 6 values has a denoising effect, at least for these 18 features.

The distribution among the 4 movement tasks is the following : 22 features are extracted from PT task, 6 from KT and 2 from FT tasks. As for the normal data, the majority of these features are extracted from tasks based on tremors (PT, KT tasks). However, the feature with the largest coefficient correlation (0.52) is now a feature extracted from the FT task.

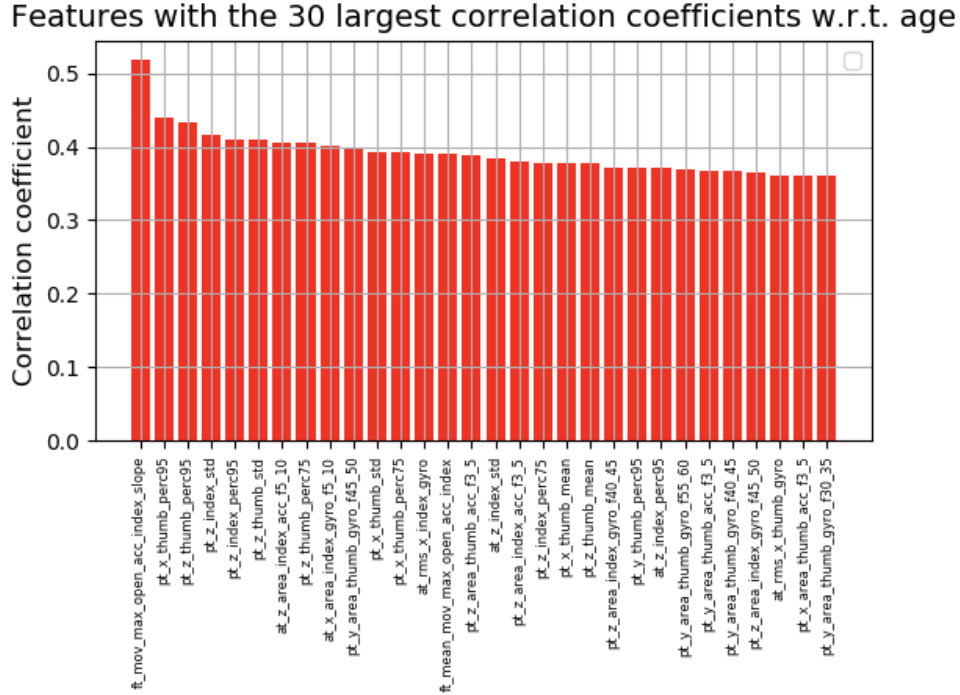


Figure 6.5: 30 features with the highest coefficient correlation w.r.t the age for the "denoised" data.

In order to analyse the redundancy of the features of the denoised data, correlation coefficients between each pair of features have been computed. The correlation coefficient matrix is shown as a heatmap on Figure 6.6. This heatmap is quite similar to the one produced with normal data: each feature seems to be highly correlated with the features of the same task, and the feature of the FT task are also highly correlated with the ones of the PS task.

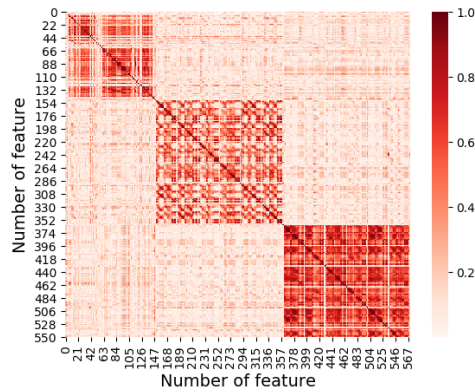


Figure 6.6: Heatmap of the correation coefficient matrix for the "denoised" data.

Finally, the plot representing the first principal component of these denoised data according to the second is shown on Figure 6.7. Since there are fewer samples, there are fewer points than on the plot of the PCA of the normal data, but as on the other plot, no cluster or trend can be seen.

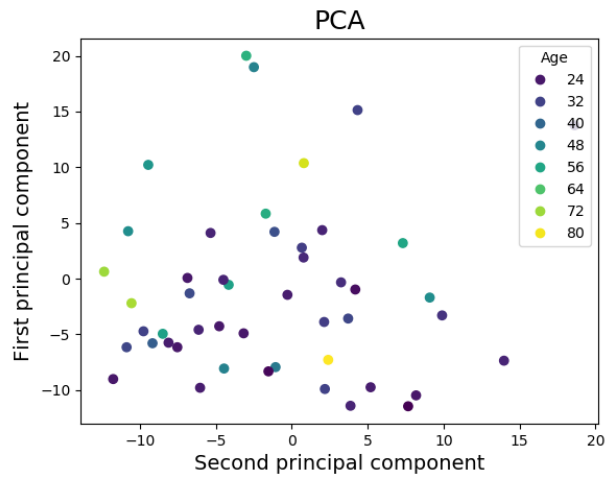


Figure 6.7: First principal component according to the second with colors representing the age for the "denoised" data.

Chapter 7

Discriminative models for classification tasks

In this chapter, an introduction to the machine learning models used for predicting age ranges and repetition numbers is given. Their advantages and weak points are also discussed.

7.1 Formalisation of the supervised classification tasks

The two main goals are to predict age range and repetition number from the data set composed of volunteer data and features extracted from the records. On the one hand, for age range prediction, the ages are removed from the features, transformed into numbers representing the age ranges and used as label data. On the other hand, for the repetition number prediction, it is the number of the repetition (0,1,2) which is removed from the features and used as such as a label. In the two cases, labels are known, used to build the models, and are discrete, which leads us to use models for supervised classification tasks.

Each sample is represented by a vector $\mathbf{x} \in \mathbb{R}^d$ composed of d features. For each sample, a scalar y representing the age range prediction or the repetition number prediction is assigned.

For the age range prediction,

$$y = \begin{cases} 0 & \text{if } 18 \leq \text{age} \leq 41 \text{ years} \\ 1 & \text{if } 42 \leq \text{age} \leq 64 \text{ years} \\ 2 & \text{if } 65 \leq \text{age} \leq 87 \text{ years} \end{cases} \quad (7.1)$$

For the repetition number prediction, $y = 0, 1$ or 2 for the first, second and third task's repetition.

7.2 Support Vector Machine

The SVM is an algorithm that aims to solve a binary classification problem and can be extended to multi-class problems. The goal of the SVM is to find an optimal hyperplane from the set of existing hyperplanes that can separate the data in the input space or in a higher-dimension space. The left graph on Figure 7.1 shows the possible separations of data points with blue points having target class Blue and red points having target class Red. Looking at the right graph on Figure 7.1, the decision boundary represents the optimal hyperplane and the two dotted lines represent some other hyperplanes passing through the data-points closest to the hyperplane. These data-points are known as support vectors and the distance between them and the optimal hyperplane is the margin. Maximizing this margin allows the SVM to find the optimal hyperplane.

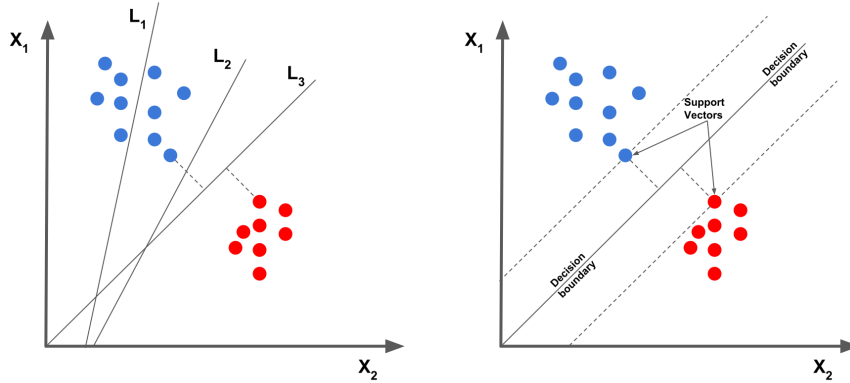


Figure 7.1: 2D examples of hyperplanes. Left: Multiple separating hyperplanes L_i with different margins. Right: Maximal margin hyperplane with highlighted support vectors.[25]

The distance r from any point to the hyperplane can be computed from $r = \frac{g(\mathbf{x})}{\|\mathbf{w}\|}$ where $g(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + w_0$ is the hyperplane where $\langle ., . \rangle$ represents the dot product between two vectors. When $\mathbf{x}$ is a support vector, the distance r is called the geometrical margin. This geometrical margin is then defined as $\frac{1}{\|\mathbf{w}\|}$ since a fix margin $|g(\mathbf{x})| = 1$ is chosen for support vectors, called the functional margin.

Maximizing this margin can be done by minimizing $\|\mathbf{w}\|$ or equivalently $\frac{\|\mathbf{w}\|^2}{2}$.

Hard margin SVM are the most common form of SVM and search for a separating hyperplane under the assumption that the data is linearly separable either with the data as is or through a **kernel trick** which is covered in details later. This separating hyperplane classifies perfectly the training data and has a functional margin of 1, hence the name hard margin as it is a fixed size margin. This hyperplane is defined by a weight vector $\mathbf{w}$ and a scalar bias w_0 where the constraint of Equation 7.2 must be met.

$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + w_0) \geq 1, \forall i = 1, \dots, n \quad (7.2)$$

Minimizing $\frac{\|\mathbf{w}\|^2}{2}$ under this constraint is a constrained optimization problem and can be tackled by transforming it into a Lagrangian optimization problem L. The goal is to minimize this Lagrangian problem (Equation 7.3) with respect to primal variables $\mathbf{w}, w_0$ as it is the original optimization problem to solve.

$$L(\mathbf{w}, w_0, \boldsymbol{\alpha}) = \frac{\|\mathbf{w}\|^2}{2} - \sum_{i=1}^n \alpha_i (y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + w_0) - 1) \quad (7.3)$$

where L is the Lagrangian optimization problem (Primal problem) and α_i are the Lagrange multipliers.

Additionally the Lagrangian has to be maximized with respect to dual variables α_i . Given primal $\mathbf{w}, w_0$ and dual $\boldsymbol{\alpha}$, the duality gap between these two problems is measured by the quantity $f_p(\mathbf{w}, w_0) - f_d(\boldsymbol{\alpha})$ where f_p is the primal problem and f_d the dual problem. The point where the duality gap is zero is called the saddle point and occurs when $\mathbf{w}, w_0$ are primal optimal (similarly $\boldsymbol{\alpha}$ is dual optimal). At this saddle point, the derivatives of L can be computed with respect to primal variables w and w_0 resulting in $\sum_{i=1}^n \alpha_i y_i = 0$ and $\mathbf{w} = \sum_{i=1}^n \alpha_i y_i \mathbf{x}_i$. Given these two formulas, the optimization problem can now be rewritten to remove the original primal variables and gives the dual optimization problem written in Equation 7.4.

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,i'=1}^n \alpha_i \alpha_{i'} y_i y_{i'} \langle \mathbf{x}_i, \mathbf{x}_{i'} \rangle \quad (7.4)$$

where this dual problem is solvable under the condition that $\alpha_i \geq 0$ and $\sum_{i=1}^n \alpha_i y_i = 0$.

For the solution of the optimization problem to be optimal, it has to satisfy the Karush-Kuhn-Tucker conditions. The complementary slackness conditions is such

that $\alpha_i \cdot h_i(\mathbf{x}) = 0, \forall i = 1, \dots, n$ where α_i is a dual variable and $h_i(\mathbf{x})$ the primal constraint function. Substituting the variables in this condition gives the Equation 7.5.

$$\alpha_i \underbrace{(y_i(< \mathbf{w}, \mathbf{x}_i > + w_0) - 1)}_{h_i(\mathbf{x})} = 0, \forall i = 1, \dots, n \quad (7.5)$$

where the constraint can be satisfied as a strict inequality as a way to maximize L (Equation 7.6a) or satisfied as an equality in which case the data-point x_i is a support vector (Equation 7.6b).

$$\alpha_i = 0 \iff y_i(< \mathbf{w}, \mathbf{x}_i > + w_0) > 1 \quad (7.6a)$$

$$\alpha_i > 0 \iff y_i(< \mathbf{w}, \mathbf{x}_i > + w_0) = 1 \quad (7.6b)$$

The decision function classifying new data-points can now be written as Equation 7.7.

$$\begin{aligned} f(\mathbf{x}) &= \text{sign} [< \mathbf{w}, \mathbf{x} > + w_0] \\ &= \text{sign} \left[\sum_{\mathbf{x}_i \in SV} \alpha_i y_i < \mathbf{x}_i, \mathbf{x} > + w_0 \right] \end{aligned} \quad (7.7)$$

where SV is the set of Support Vectors (SV) from the training samples, α_i the solutions to the dual problem and w_0 that can be computed with any support vector through $y_i(< \mathbf{w}, \mathbf{x}_i > + w_0) = 1$.

Multi-class classification can be done in two ways: "one-against-one" and "one-vs-the-rest". For the One-against-one scheme the model constructs $n_class * (n_class - 1)/2$ classifiers where n_class is the number of target classes and each one is trained on data containing only two classes. A One-versus-the-rest scheme trains only one classifier per class. For each classifier, the data separates the data in two classes, the target class and a merged group containing the other classes.

Kernels

Unfortunately, data linearity cannot be guaranteed and could thus lead to data-points being not linearly separable with margin 1 in the input space. This problem can be solved with the previously mentioned **kernel trick**. A kernel k is a symmetric function $k(\mathbf{x}, \mathbf{x}') = < \phi(\mathbf{x}), \phi(\mathbf{x}') > = k(\mathbf{x}', \mathbf{x})$ where $\phi(\cdot)$ defines a non-linear

mapping to a new feature space.

The Mercer conditions for a symmetric function k are that $k : X \times X \rightarrow \mathbb{R}$ is a kernel if for any finite subset $\mathbf{x}_1, \dots, \mathbf{x}_n$ of X the gram matrix $K = [k(\mathbf{x}_i, \mathbf{x}_{i'})]_{i,i'=1}^n$ is positive semi-definite, meaning that the eigenvalues must be non-negative.

Given a symmetric function k that satisfies these conditions, there exists a mapping ϕ such that $k(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle$. This kernel trick can now be used by the SVM since it uses dot products on the input data. $\langle \mathbf{x}, \mathbf{x}' \rangle$ is then replaced by $k(\mathbf{x}, \mathbf{x}')$ such that the decision function becomes Equation 7.8.

$$f(\mathbf{x}) = \text{sign} \left[\sum_{\mathbf{x}_i \in SV} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) + w_0 \right] \quad (7.8)$$

Predefined kernels exist and can be used as is. We present the most well-known kernels implemented, e.g. the SVM kernels from *scikit-learn*:

- The **Linear kernel** maps the data to itself.
- The **Polynomial kernel** maps the data to $(\gamma \langle \mathbf{x}, \mathbf{x}' \rangle + r)^d$ where γ is the coefficient of influence of a single training sample, d controls the degree of the polynomial function and r is the regularization parameter. γ must be strictly positive, d must be a natural number and r must be a positive real number. r is a constant that trades off the correct classification of training samples against the maximization of the decision function. A larger value of r implies that a smaller margin is going to be accepted if all training data-points can be classified correctly. On the other side, a larger margin allows a simpler decision function but this comes at the cost of training accuracy.
- The **Radial Basis Function kernel** or shorter RBF kernel maps the data to $\exp(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2)$ where γ is the inverse of the variance of the support vectors.
- The **Sigmoid kernel** maps the input data to $\tanh(\gamma \langle \mathbf{x}, \mathbf{x}' \rangle + r)$. The γ and r parameters are the same as for the polynomial kernel.

Advantages and disadvantages of SVM

Computing the optimization problem can take time but finding a small set of support vectors is computationally efficient because any new unseen examples can then be classified using only the data from the support vectors. Moreover SVM is robust, any non-support vector data-point that would be removed from the

training samples would leave the the solution vector $\mathbf{w}$ unchanged and the removed data-point would still be classified correctly if presented as unseen data-points. Nevertheless in order to build a good model the kernel given that the number of features is greater than the number of samples, kernel hyperparameters and the regularization constant C have to be chosen very carefully[1].

7.3 Multi-layer Perceptron

The MLP, also called feedforward neural network, is a supervised learning algorithm able to approximate non linear functions. It forms the basis architecture of deep learning algorithms and aims to find the best mapping $\mathbf{y} = f(\mathbf{x}; \boldsymbol{\theta})$ between an input $\mathbf{x}$ and the label(s) $\mathbf{y}$ of this input, by finding the parameters $\boldsymbol{\theta}$ through gradient-based learning using back-propagation.

7.3.1 Structure

An example of a MLP architecture can be found on Figure 7.2. Multi-layer perceptrons are made of units also called neurons (gray circles in Figure 7.2). They are organised into at least three layers : an input layer, one or more hidden layers (two in the example of Figure 7.2) and an output layer. Each input value is an element of the input vector $\mathbf{x}$. Multi-layer perceptrons can have multiple outputs (three in the example of Figure 7.2) and may thus be represented by a vector. But in this case, the output is a value representing the age range or the repetition number and thus is represented by the scalar y . A hidden layer is represented by a vector where each element is the value of a neuron that is present in this particular layer.

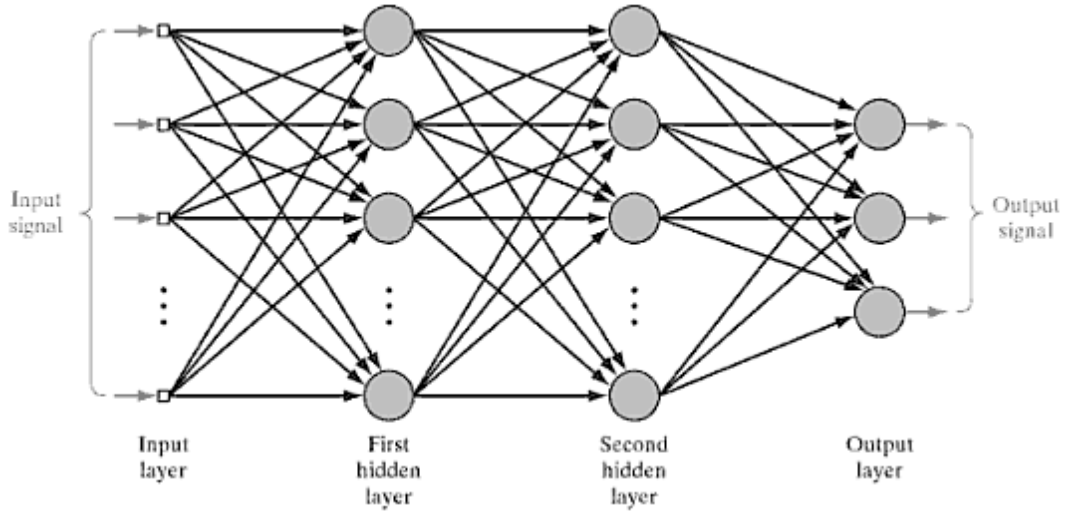


Figure 7.2: Structure of a multi-layer perceptron : one input layer, at least one hidden layer (two in this example) and one output layer.

For a multi-layer perceptron with k hidden layers, inputs $\mathbf{x}$ and hidden layers $\mathbf{h}^{(1)}, \dots, \mathbf{h}^{(k)}$ are linked by the Equation 7.9a and 7.9b.

$$\mathbf{h}^{(1)} = g^{(1)}(\mathbf{W}^{(1)T}\mathbf{x} + \mathbf{b}^{(1)}) \quad (7.9a)$$

$$\mathbf{h}^{(j)} = g^{(j)}(\mathbf{W}^{(j)T}\mathbf{h}^{(j-1)} + \mathbf{b}^{(j)}), \quad 1 \leq j \leq k \quad (7.9b)$$

Where $\mathbf{h}^{(j)}$ is j -th hidden layer, $g^{(j)}$ is an activation function of layer j , $\mathbf{W}^{(j)}$ is a weight matrix connecting layer $j-1$ to layer j and $\mathbf{b}^{(j)}$ is a bias vector of layer j and . Each element $w_{r,s}$ of a matrix $\mathbf{W}^{(j)}$ is a weight (scalar value) multiplying neuron r of the the hidden layer $j-1$ to participate to the value of neuron s in the hidden layer j .

The probability that the sample $\mathbf{x}$ has the output label c is computed in Equation 7.10.

$$\hat{P}(y = c|\mathbf{x}) = softmax(\mathbf{z})_i = \frac{\exp(z_i)}{\sum_{l=1}^k \exp(z_l)} \quad (7.10)$$

where $\mathbf{z} = \mathbf{W}^{(k+1)T}\mathbf{h}^{(k)} + \mathbf{b}^{(k+1)}$ and $z_i = \log P(y = i|\mathbf{x})$ The predicted class $\hat{y}$ is the one with the highest probability, thus $\hat{y} = argmax_i \hat{P}(y = i|\mathbf{x})$.

It is the use of non-linear activation functions that allows networks to be able to approximate functions for non-linear separable data. Many types of activation

functions exist but among them, Table 7.1 shows some of the most popular activation functions.

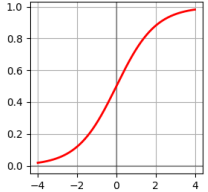
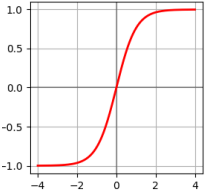
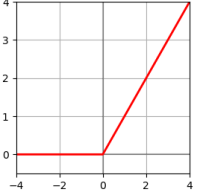
Function	logistic or sigmoid	tanh	ReLU
Equation	$f(z) = \frac{1}{1+\exp(-z)}$	$f(z) = \tanh(z)$ $= \frac{\exp(z)-\exp(-z)}{\exp(z)+\exp(-z)}$	$f(z) = \max(0, z)$
Plot			

Table 7.1: Comparison of some of the most popular activation functions. Logistic and hyperbolic tangent functions were the most used before the emergence of the Rectified Linear Unit (ReLU) function, which is now considered as the best default choice.

Logistic and hyperbolic tangent functions were the most used before the emergence of the Rectified Linear Unit (ReLU) function. However, they have the disadvantage to "saturate" for large positive values and negative values : the gradient becomes vanishingly small and the convergence to good weights becomes thus very slow. This problem is known as the "vanishing gradient problem". The hyperbolic tangent is said to perform better than the sigmoid function but its use is now discouraged while the ReLU function is now considered as the best default choice.[10].

7.3.2 Learning

Now the basic structure of a multi-layer perceptron has been seen, we still need to understand how the values for the weights and biases are learned. Weights and biases are the parameters that need to be optimized and we denote them θ . This learning phase is done by using a cost function and an optimisation algorithm called stochastic gradient descent.

The cost function $J(\theta)$ of a sample $\mathbf{x}$ is in this thesis composed of a loss function $L(\hat{y}, y)$ that computes the "loss" between predicted output $\hat{y} = f(\mathbf{x}; \theta)$ and the true label y , and a regularisation parameter $\Omega(\theta)$ as shown in Equation 7.11.

$$J(\theta) = L(\hat{y}, y) + \Omega(\theta) \quad (7.11)$$

Where :

- $L(\hat{y}, y)$ is the cross entropy loss function : $L(\hat{y}, y) = -\sum_{c=1}^C y_c \log(p_c)$ where y_c is 1 if class label c is the true target class and 0 otherwise, and $p_c = \hat{P}(y = c|\mathbf{x})$, the predicted probability of class c for input $\mathbf{x}$.
- And $\Omega(\boldsymbol{\theta})$ is a L2-regularization term that aims to penalize large weights that could cause overfitting : $\Omega(\boldsymbol{\theta}) = \alpha \sum_{i=1}^l \sum_{r,s} (\mathbf{W}_{r,s}^{(i)})^2$ where α is a non-negative hyper-parameter allowing to control the regularization and $\mathbf{W}_{r,s}^{(i)}$ is the element at indices r,s of the weight matrix of layer i $\mathbf{W}^{(i)}$.

The stochastic gradient descent procedure is summarized in Algorithm 7.1¹. The key idea of this iterative algorithm is, at each iteration, to estimate the gradient $\hat{\mathbf{g}}$ of the cost function on a relatively small number of samples (minibatch) drawn uniformly from the training set and to use the estimated gradient downhill to update the values of the parameters so that the value of the cost function is decreased. In this algorithm, ϵ_k is a learning rate whose value can decrease with iterations and is an important hyper-parameter of the algorithm. The initial values of the parameters are small random numbers. Note that this algorithm considers a general vector value $\mathbf{y}$ for the targets but in this case, it is a scalar.

Algorithm 7.1 Stochastic gradient descent (SGD) update

Require: Learning rate schedule ϵ_1, ϵ_2

Require: Initial parameter $\boldsymbol{\theta}$

$k \leftarrow 1$

while Stopping criterion not met **do**

 Sample a minibatch of m examples from the training set $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}\}$ with corresponding targets $\mathbf{y}^{(i)}$

 Compute gradient estimate $\hat{\mathbf{g}} \leftarrow \frac{1}{m} \nabla_{\boldsymbol{\theta}} \sum_i L(f(\mathbf{x}^{(i)}; \boldsymbol{\theta}), \mathbf{y}^{(i)})$

 Apply update: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \epsilon_k \hat{\mathbf{g}}$

$k \leftarrow k + 1$

end while

As we can see in Algorithm 7.1, the gradient of the cost function is used but the way to compute it is not explained. Once the information of the inputs is propagated forward through the network to be able to compute the cost function, we have to back-propagate the information through the network to be able to update weights and biases of all layers. This process of backward propagation is

¹This algorithm comes from the book "Deep Learning" of Goodfellow et al.2016[10]

described in Algorithm 7.2². As for the first algorithm, true targets $\mathbf{y}$ and predicted targets $\hat{\mathbf{y}}$ are generalized to vectors and in this algorithm, $\mathbf{a}^{(k)} = \mathbf{W}^{(k)}\mathbf{h}^{(k-1)} + \mathbf{b}^{(k)}$ and f is an activation function.

Algorithm 7.2 Backward propagation

```

After the forward computation, compute the gradient on the output layer
 $\mathbf{g} \leftarrow \nabla_{\hat{\mathbf{y}}} J = \nabla_{\hat{\mathbf{y}}} L(\hat{\mathbf{y}}, \mathbf{y})$ 
Backward propagation
for  $k = l-1, \dots, 1$  do
    Convert the gradient on the layer's output into a gradient on the pre-
    nonlinearity activation
     $\mathbf{g} \leftarrow \nabla_{\mathbf{a}^{(k)}} J = \mathbf{g} \odot f'(\mathbf{a}^{(k)})$ 
    Compute gradient on weights and biases
     $\nabla_{\mathbf{b}^{(k)}} J = \mathbf{g} + \lambda \nabla_{\mathbf{b}^{(k)}} \Omega(\boldsymbol{\theta})$ 
    Propagate gradients w.r.t. the next lower-level hidden layer's activations:
     $\mathbf{g} \leftarrow \nabla_{\mathbf{h}^{(k-1)}} J = \mathbf{W}^{(k)T} \mathbf{g}$ 
end for

```

Finally, let us add that Sklearn provides an implementation (*neural_network.MLPClassifier*) of the multi-layer perceptron described above and it is this implementation we use to predict both the repetition number and the age.

7.3.3 Advantages and disadvantages of MLP

The multi-layer perceptron has a good theoretical motivation : according to the *universal approximation theorem*, a multi-layer perceptron with a linear output layer, an activation function such as the sigmoid function and provided that there are enough hidden units, can approximate any continuous function on a closed and bounded subset of $\mathbb{R}^n$ from a finite-dimensional space to another with any desired non zero amount of error[10].

However, it has also several drawbacks, like the large number of hyper-parameters to choose (the structure of the network, the activation function, the learning rate, the regularisation parameter, etc.), and the computational cost of the training can be very expensive. The choice of the hyper-parameters will be discussed in the chapter 9.

²As Algorithm 7.1, Algorithm 7.2 also comes from the book "Deep Learning" of Goodfellow et al.2016[10]

Chapter 8

Multi-dataset and multi-class performance assessment

When building predictive models, it is always important to keep in mind how well the trained model might perform on unseen data. This unseen data is not always available and might make it hard to have a model making accurate predictions. Moreover, training a model and evaluating it on a part of the data that the model has already seen is a common problem in machine learning and creates an optimistic bias. The predictions made by the model are very likely to be better than predictions performed on unseen data and are biased as the model has already seen the data. Making predictions on real unseen data with such a model could result in huge differences in performances. This chapter covers the use of hold-out testing and cross-validation to remove the optimistic bias from the performances.

8.1 Hold-out testing

A very simple yet sometimes costly technique, especially when the data is sparse, is called hold-out testing. This technique consists of splitting the data set in two parts: a train set and a test set. The model is then trained on the train set and its performance is evaluated on the test set. Common splits used in machine learning for this technique are either to keep 80% or 90% of the data for training and respectively use the remaining 20% or 10% of the data for testing.

8.1.1 Data leakage

Randomly splitting the data set taking only into account the split percentages would introduce a bias in the sets. As a reminder, the data consists of 6 samples for each volunteer. If a part of these samples makes it into the test set, there might be what is called data leakage. This phenomenon occurs when information is shared between the train set and the test set. For example the model would learn on 4 of the 6 samples of one volunteer. When making a prediction on the 2 remaining samples, data of the 4 first samples could be used to predict the target class. The evaluation of this model would be misleading as its performances on real unseen data could be much lower than expected. For this problem to be avoided, the solution is to split based on the volunteers. When splitting, we give the data splitter information based on which it can keep samples grouped either in the train set or the test set. This is managed through what are called "groups" where one group represents one volunteer and is applied both on the train-test split as well as the k-fold cross-validation.

8.1.2 Data imbalance

Another problem that arises for the age range prediction is data imbalance. As mentioned in section 7.1, the age feature has been reorganised in three classes when predicting the age:

- ages ranging from 18 until 41 representing class 0: 240 samples,
- ages ranging from 42 until 64 representing class 1: 84 samples,
- ages ranging from 65 until 87 representing class 2: 30 samples.

If this imbalance isn't taken into account when creating the train and test set, a test set containing only samples from the majority class would lead to very high prediction performances because the model would have learned to predict the majority class most often. Trying to predict on unseen data from minority classes would show very low performances. When splitting data, there is no guarantee that the class balance remains the same. Stratified sampling when making the train-test and train-validation splitting is one of the ways around this problem in machine learning. Stratified sampling takes a subsample with a different size out of each class in order to keep the original class balance.

This problem is absent when predicting the repetition number of a record since all the volunteers performed the tests three times. Thus, the distribution of these repetition numbers is balanced.

8.1.3 Data resampling

By applying all the techniques mentioned before, a good grasp on how the data is split in train and test sets can already be gotten. However, basing the performance of a model on one train-test split will only give information about how the model might perform for some volunteers, those who have been selected in the test set. The performances could differ with other volunteers in the test set. To solve this, we perform multiple resamples, i.e. we make the train and test set vary a high number of times and collect the performances of the model on these different resamplings to get the average performance of our model. For this purpose, 200 resamples were computed.

8.2 Cross-validation

In addition to finding good models, values for the hyper-parameters of each model also have to be chosen and these hyper-parameters need to be assessed to find which values give the best results. But the same samples to train the model and to choose the hyper-parameter values cannot be used, it could lead to another optimistic bias. In order to avoid it, a K-fold cross validation scheme can be used. It allows to split the train set data into multiple folds where one fold contains a part of the train data that will be used as test set to validate the model. Each fold is used exactly one time as test set and is part of the train set the rest of the time. The amount of folds has to be determined beforehand and typical values for k are 5 and 10 where the percentage of data used as test set is respectively 20% and 10%. All the folds have the same size.

8.3 Splitting with attention to groups and class balance for the age range prediction

Combining the information from the groups and the class balance results in a complex splitting context. Whether it is when splitting the data between a train and test set or when performing cross-validation, both factors have to be taken into account. This means that the train, validation and test sets always have the same proportions of classes and the samples from one volunteer are always grouped together. An example of train-test split for the age range prediction task is given in Table 8.1.

	Train	Test
ID	1 2 3 4 5 6 7 8 10 11 12 14 15 17 19 20 21 23 24 25 26 27 28 30 31 32 33 34 35 37 39 41 43 44 45 46 47 49 50 51 52 53 54 55 56 57 59	9 13 16 18 22 29 36 38 40 42 48 58
volunteers	47 (79,6%)	12 (20,4%)
classes	0: 192 (68,09%) 1: 66 (23,4%) 2: 24 (8,51%)	0: 48 (66,67%) 1: 18 (25%) 2: 6 (8,33%)

Table 8.1: Train-test split with attention to groups and class balance

The splitter keeps approximately the same proportion of classes in each set. The ID row contains all the unique volunteer ids in each set. It can be noticed that no id is present in both sets which indicates the split took this variable into account. The size of the test set is also respected as we set a value of 20% for this example split.

8.4 Performance metrics

For cross-validation, an evaluation metric has to be chosen. This metric guides the cross-validation when searching for the best hyper-parameters for one model and is thus very important. As the age data is imbalanced, we have chosen to evaluate age prediction models on the balanced accuracy score written in Equation 8.1.

$$Balanced\ accuracy\ score = \frac{1}{n_class} \sum_{i=1}^n recall_i \quad (8.1)$$

where $recall_i$ is the recall for class i , the number of true positives divided by the number of true positives + the number of false negatives. This represents the proportion of correctly classified samples with true class i among all the samples with class i (correctly classified as i or misclassified).

This metric is typically used in multi-class classifications tasks to deal with imbalanced data sets as it allows models to maximize the class balance accuracy during training. This method yields in general higher test set accuracy and recall than other existing methods[18]. The same metric is used for the prediction of the

repetition number as for balanced data sets, the balanced accuracy score and the accuracy score are identical.

8.5 Multi-classifiers and multi-dataset performance assessment

Comparing performances of multiple algorithms on one data set is a common step in machine learning projects. When it comes to the performances of algorithms on multiple data sets (or different angles of view of one data set), the Friedman test proves to be very useful when followed by a post-hoc test[7]. The first test checks whether there is a difference between k (>2) classifiers with the null-hypothesis being that there is no difference between these k classifiers. This is followed by a post-hoc test that determines where the differences lie.

8.5.1 Friedman test

This description of the Friedman test is based on the approach of Demsar[7]. In the Friedman test, several algorithms are ranked according to their performances compared to each other. The algorithms are all ranked for each data set separately where the algorithm with the best performance gets the rank 1 followed by rank 2 for the second best, 3 for the third best, etc. The Friedman test compares the average ranks of the algorithms under the null-hypothesis. This hypothesis states that the ranks of all the algorithms should be equal. Let k be the total number of algorithms to compare and N the total number of data sets used to rate the performances of the algorithms. The Friedman statistic is computed in Equation 8.2.

$$\chi_F^2 = \frac{12N}{k(k+1)} \left[\sum_{j=1}^k R_j^2 - \frac{k(k+1)^2}{4} \right] \quad (8.2)$$

where $R_j = \frac{1}{N} \sum_i r_i^j$ is the average rank of algorithm j . r_i^j is the rank of the j -th algorithm on the i -th data set.

The Friedman statistic is distributed according to χ_F^2 with $k - 1$ degrees of freedom, when N and k are big enough. For smaller values of N and k ($N < 10$ and $k < 5$), exact critical values have been computed. Iman and Davenport derived an improved statistic distributed according to the Fisher-distribution with $k - 1$

and $(k - 1) \times (N - 1)$ degrees of freedom shown in Equation 8.3.

$$F_F = \frac{(N - 1)\chi_F^2}{N(k - 1) - \chi_F^2} \quad (8.3)$$

The statistical tests have to be performed with the new critical values of the F -distribution. If the null-hypothesis is rejected with a confidence of α (typical values are 0.01, 0.05 and 0.1), a post-hoc test can be performed.

8.5.2 Post-hoc tests

Given that the null-hypothesis of the Friedman test (no significant difference between the k classifiers) has been rejected, further analysis is now possible using a post-hoc test. This subsection covers two types of post-hoc tests: the Nemenyi test[19] and the Bonferroni-Dunn test[9].

Nemenyi test

When all classifiers are compared to each other, the Nemenyi test gives an indication of the difference significance between two classifiers for all the classifiers. This difference is significant when the ranks of the algorithms differ by at least what is called the critical difference. This critical difference can be computed as shown in Equation 8.4.

$$CD = q_\alpha \sqrt{\frac{k(k + 1)}{6N}} \quad (8.4)$$

where q_α is the critical value of the Student distribution for infinite degrees of freedom, k comparisons and an α value (0.10, .05, .025, 01 or .005) divided by $\sqrt{2}$. An example of such critical values can be found in Table 8.2. For example, the critical value for 3 classifiers is computed by taking the two-tailed critical value of the Student distribution for infinite degrees of freedom, an alpha value of 0.05 and 3 comparisons, i.e. 3.314. Dividing this value by $\sqrt{2}$ gives 2.343, the first value of the table.

#classifiers	3	4	5	6	7	8	9	10
$q_{0.05}$	2.343	2.569	2.728	2.850	2.949	3.031	3.102	3.164

Table 8.2: Critical values for the two-tailed Nemenyi test

Now that the critical difference is computed, it can be used to compare the algorithms. If the average rank difference between two algorithms is smaller than

the critical difference, it means the existence of a significant difference between the two algorithms can't be determined. The opposite is proven if the average rank difference exceeds the critical difference.

Bonferroni-Dunn test

The Bonferroni-Dunn test uses a control classifier to compare the performances of all the algorithms. This test is part of the procedures that control the family-wise error in multiple hypothesis testing. This test uses the same computation of the critical difference as the Nemenyi test except that α is divided by $k - 1$ for the critical values q_α . Some critical values for $\alpha = 0.05$ can be found in Table 8.3. For example, the critical value for 3 classifiers is computed by taking the two-tailed critical value of the Student distribution for infinite degrees of freedom, an alpha value of $\frac{0.05}{3-1} = 0,0167$ and 2 comparisons, 2.241.

#classifiers	3	4	5	6	7	8	9	10
$q_{0.05}$	2.241	2.394	2.498	2.576	2.638	2.690	2.724	2.773

Table 8.3: Critical values for the two-tailed Bonferroni-Dunn test

As for Nemenyi, comparisons can now be made based on the computed CD. The difference between the average rank of an algorithm and the average rank of the control algorithm allows to tell that there exists a significant difference between these two algorithms if their average rank difference is bigger than the critical difference.

Chapter 9

Experimental results

This chapter addresses the methods used to compare machine learning algorithms and to study the training effect and the influence of the age on the four selected tasks of the motor examination part of MDS-UPDRS : FT, PS, PT, KT. With the gathered results, an answer is given to the three questions raised in this study:

- What are the best machine learning approaches to analyze these movement data ?
- What is the influence of the age on the quality of these movements ?
- How is the training effect marked on these movements ?

9.1 Data sets

To compare the chosen algorithms, three versions of the data set are used:

- Ages: this data set has the age feature as its target variable and contains the extracted features of all records for each volunteer.
- Ages DN: the Ages Denoised data set is the same data set as the Ages data set except that for each volunteer, all repetitions of a task are summarized in one by taking the median of all values of the features. As explained in 6.4, this leaves 59 samples instead of 354.
- Repetition: this data set uses the repetition number as its target variable and all the other features of the volunteers as predictor variables.

9.2 Experimental settings

This section describes the process of carrying out the experiments, e.g. the choice of hyperparameters of the models and the choice of parameters to assess the performances (percentage of training and test sets, the number of resamples, etc.).

For some volunteers, there are features that can't be extracted due to problems for one task or for one method. This creates empty values in the data set that are filled to avoid having to delete the entire row. We have chosen to perform data imputation using the median value of a feature column to fill in these empty values.

For the results to be robust, we have trained and tested each model on 200 resamples. In this way, the variability obtained in the different test metrics decrements and the measured mean $\bar{\mu}$ of the performances approximates the real mean μ of the performances of the algorithms.

We have performed the train-test splits by keeping 20% of the data as test set for each resample. As mentioned in the different sections of chapter 8, each set contains approximately the same proportion of classes and all the data records of one volunteer are either in the train set or in the test set but not in both.

As seen in chapter 6, the data contains some outliers. Knowing this, we have used a robust scaler that removes the median and scales the data according to interquartile range. The test set is scaled using the values learned by the scaler of the train set. Scaling the data set after train-test split helps to avoid any data leakage as the data set that would be scaled on the complete data set would introduce some information of the train set into the test set.

Furthermore, PCA has been applied to reduce the number of features of both training set and test set. A high percentage of variance was kept while reducing the number of features. We kept 99 % of the total variance in the new components since it offered a good compromise between number of features and performance. PCA was fitted on the train set and the fitted PCA model has then been used to transform both the train set and the test set.

Both for age and repetition number prediction, the performances of the following classifiers have been compared: a SVM with default hyperparameters and two other SVM (one using a linear kernel and the other a rbf kernel) with some hyperparameters tuned for the optimal balanced accuracy, a MLP with default hyper-parameters, and two other MLP (one with the tanh activation function, the other with the ReLU activation function) with some hyperparameters tuned for

the optimal balanced accuracy. More details about the hyperparameters are given below.

Moreover, as a part of the model comparison, we have chosen to incorporate a dummy or random classifier. This classifier serves as a base classifier and is configured to make a prediction based on the class balance of the train set that it is given. As this model predicts an output randomly, it is used as a baseline classifier to compare whether the other models perform better than this baseline classifier.

The values of the hyperparameters of all the classifiers, except for the Random, default SVM and default MLP classifiers, have been grid-searched using a k-fold cross-validation with k set to 5. This value of k has been set to have 80% of the train set used for training and 20% of the train set used for validation. The set of hyper-parameters didn't change between each resample and the best hyper-parameters have been chosen based on the model returning the highest balanced accuracy score on the validation set.

Finally, a given value has been fixed for the random state of the train-test split function. This enables to use the same splits in train and test set for the two data set versions for the age range prediction task.

9.3 Model hyperparameters

For each classifier, a fixed set of hyperparameters is defined. These sets of hyperparameters are the same for all the data sets. These hyperparameters are listed here.

- **Random classifier.** This classifier corresponds to a random model that is configured to take into account the class balance.
- **SVM classifier.** The default SVM classifier with a RBF kernel, a regularization parameter $C = 1$ and a gamma $\gamma = \frac{1}{n_features * X.var()}$ where $n_features$ represents the number of features of the data set and $X.var()$ is the variance of the data set.
- **SVM+1 classifier.** An SVM with a linear kernel. The regularization parameter is grid-searched with the following range of values: [0.0001, 0.001, 0.01, 0.1, 1, 10]. Moreover, an additional parameter present in the SVM implemented by Scikit-learn allows to control how the decision function handles the multi-class case: either by using a one versus one scheme or the one versus the rest scheme. We have chosen to keep the one versus rest

scheme as it is the default option. Finally, when performing age prediction, an additional hyperparameter is used that ensures that the regularization parameter is adapted for each class according to its class weight, i.e. the inverse of the number of samples with this class. This allows the classifier to take into account the class balance when training and will penalize misclassifications harder the lower the number of samples with this class is.

- **SVM+r classifier.** An SVM using a RBF kernel. All the other hyperparameters use the same range as specified for the SVM with linear kernel.
- **MLP classifier.** The MLP classifier with default hyperparameters is composed of one hidden layer with 100 hidden units and uses a relu activation function. As explained in the section 7.1 chapter, the classic stochastic gradient descent solver is used. In addition to this, the alpha value is fixed to 0.0001 and the initial learning rate is fixed to 0.001 (default values of the Scikit-learn implementation).
- **MLP+t classifier.** For this MLP classifier, the activation function of the hidden layers was fixed to the hyperbolic tangent (tanh) activation function. The alpha hyper-parameter acts as a regularization term and the tested values are [0.0001, 0.001, 0.01, 0.1]. At last, the initial learning rate is tested for the values 0.001 and 0.01 where the first is the default value and the latter a value determined to accelerate the learning process.
- **MLP+r classifier.** The same MLP classifier as the one presented above except for the activation function which uses the relu function now. All the other hyperparameters and their values to test are the same.

9.4 Comparison of classifiers performances

In section 8.5 about multi-dataset performance analysis of the Performance assessment chapter, the Friedman test is described as a method to compare classifiers on multiple data sets. We have chosen two different approaches to compare the classifiers. The first approach compares the algorithms by considering as different data sets the three viewpoints mentioned in section 9.1 and by taking as performance measures, for each data set, the average of the balanced accuracy of the 200 resamples found for the prediction task on that data set. This approach makes it possible to compare different models for the type of data used in this thesis, regardless of the prediction task. The second approach is a comparison for each prediction task separately : for each prediction task, all the models are compared by considering the different resamples as different data sets. This approach allows to compare the classifiers for a specific prediction task.

9.4.1 Overall comparison of classifiers

The results of the Friedman test for the chosen classifiers can be found in Table 9.1. Each row shows, for each prediction task, the average of the balanced accuracy scores computed on the 200 resamples.

	Random	SVM	SVM+l	SVM+r
Ages	0.3312 (7)	0.4038 (6)	0.4558 (2)	0.4496 (3)
Ages DN	0.3483 (7)	0.3714 (6)	0.4774 (4)	0.4526 (5)
Repetition	0.3433 (7)	0.4005 (6)	0.4208 (3)	0.4347 (1)
average rank	7	6	3	3
	MLP	MLP+t	MLP+r	
Age	0.4306 (5)	0.4696 (1)	0.4419 (4)	
Age DN	0.5016 (3)	0.5274 (1)	0.5139 (2)	
Repetition	0.4179 (5)	0.4204 (4)	0.4229 (2)	
average rank	4.333	2	2.667	

Table 9.1: Comparison of balanced accuracy scores between a random classifier, SVM with default Hyperparameters (HP), SVM tuned for linear and rbf kernel, MLP with default HP and MLP tuned for tanh and relu activation functions. The numbers in parentheses represent the ranks that are computed for the Friedman test.

On average, the MLP with a tanh activation function is the best ranked model, followed closely by the MLP with a relu activation function. The SVM with a linear kernel and a rbf kernel have rank three and are followed by the MLP with default hyperparameters with rank four. As expected, the random classifier is ranked the worst on average. Let us add that the MLP with default hyperparameters and the SVM with default hyperparameters rank respectively on average fourth and sixth. This difference seems to show that the MLP without hyperparameter optimization may perform better than an unoptimized SVM.

Computing the Friedman test as shown in Equation 9.1 allows to check whether the average ranks are significantly different from the mean rank expected under the null-hypothesis.

$$\chi_F^2 = \frac{12 \cdot 3}{7 \cdot 8} \left[(7^2 + 6^2 + 3^2 + 3^2 + 4.333^2 + 2^2 + 2.667^2) - \frac{7 \cdot 8^2}{4} \right] = 13.43 \quad (9.1)$$

Since there are 7 algorithms and 3 data sets, FF is distributed according to the F distribution with $3 - 1 = 2$ and $(3 - 1) \times (7 - 1) = 12$ degrees of freedom. The

critical value of $F(2,12)$ for $\alpha = 0.05$ is 3.88 and 2.81 for $\alpha = 0.1$. Computing the F_F value using the formula presented in section 8.5 gives the result of Equation 9.2.

$$F_F = \frac{(3 - 1) \cdot 13.43}{3(7 - 1) - 13.43} = 5.88 \quad (9.2)$$

The null-hypothesis can thus be rejected with a confidence level of 95% as $5.88 > 3.88$. Therefore, Two types of post-hoc tests can be performed: the Nemenyi test where pairwise comparisons are made and the Bonferri-Dunn test where the comparisons are made against one control variable. Both methods are covered and the results of both methods are compared.

For the Nemenyi test, the critical values can be found in Table 9.2 for alpha values of 0.05 and 0.10.

#classifiers	3	4	5	6	7	8	9	10
$q_{0.05}$	2.343	2.569	2.728	2.850	2.949	3.031	3.102	3.164
$q_{0.10}$	2.052	2.291	2.459	2.589	2.693	2.780	2.855	2.920

Table 9.2: Critical values for the two-tailed Nemenyi test

The corresponding CD for an alpha value of 0.05 with 7 classifiers is computed in Equation 9.3.

$$2.949 \cdot \sqrt{\frac{7 \cdot 8}{6 \cdot 3}} = 5.201 \quad (9.3)$$

This critical difference is bigger than the difference between the minimum average rank and the maximum average rank ($7 - 2.667 = 4.333$) which leads to Figure 9.1 below. Since this CD is bigger, all the classifier fall under the same category and their performances are thus not significantly different.

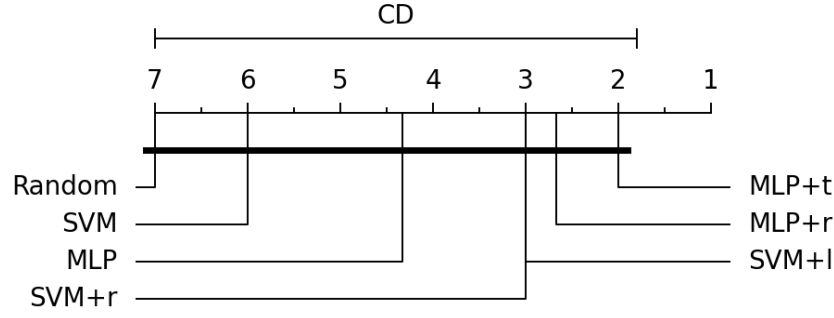


Figure 9.1: Comparison of all classifiers against each other using the Nemenyi test. Classifiers belong to the same group (connected with a black line) and their performances are not significantly different at $\alpha = 0.05$.

With an α value of 0.1, however, a difference is notable as the critical difference drops ($CD = 4.750$) as shown in Figure 9.2. This means that with a confidence level of 90%, the Nemenyi test detects a significant difference between the classifiers: the MLP+t model has significantly different performances from the baseline random classifier.

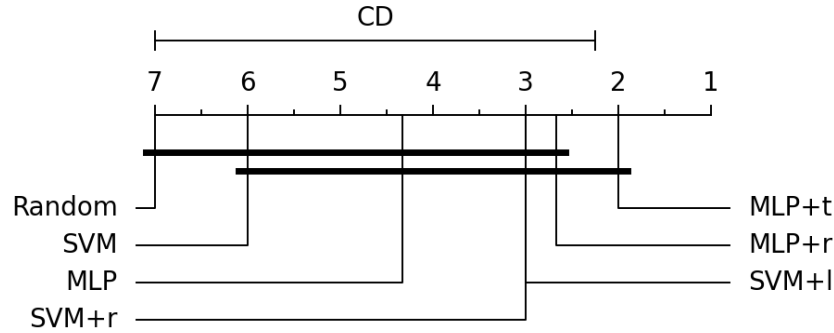


Figure 9.2: Comparison of all classifiers against each other using the Nemenyi test. Classifiers that belong to the same group (connected with a black line) are not significantly different at $\alpha = 0.1$.

As we included a control classifier (Random) in our classifiers, a Bonferroni-Dunn test can be performed to compare each classifier with the control classifier on multiple data sets. For the Bonferroni-Dunn test, the critical values can be found in Table 9.3.

#classifiers	3	4	5	6	7	8	9	10
$q_{0.05}$	2.241	2.394	2.498	2.576	2.638	2.690	2.724	2.773
$q_{0.10}$	1.960	2.128	2.241	2.326	2.394	2.450	2.498	2.539

Table 9.3: Critical values for the two-tailed Bonferroni-Dunn test

The corresponding CD for an alpha value of 0.1 with 7 classifiers is computed in Equation 9.4.

$$2.638\sqrt{\frac{7 \cdot 8}{6 \cdot 3}} = 4.223 \quad (9.4)$$

At this value of confidence, a split can be seen where two algorithms are considered as significantly different from the control classifier: MLP+t and MLP+r.

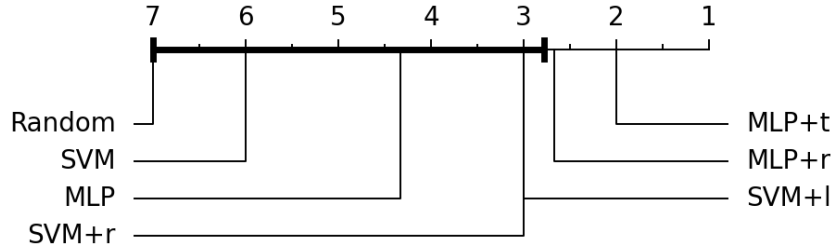


Figure 9.3: Comparison of a random classifier against the other classifiers using the Bonferroni-Dunn test. All classifiers whose rank is outside the marked interval are significantly different from the baseline random classifier at $\alpha = 0.1$.

Raising the confidence level to 95% (CD = 4.65) leaves only one algorithm that is considered as significantly different from the control classifier: MLP+t.

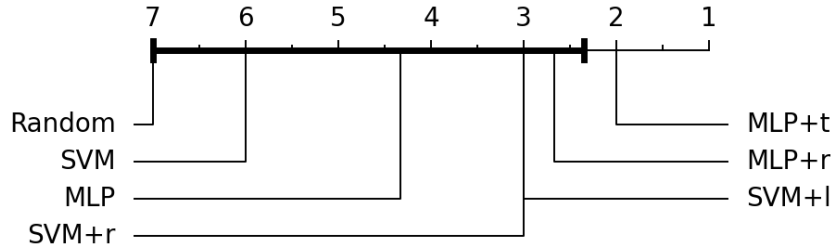


Figure 9.4: Comparison of a random classifier against the other classifiers using the Bonferroni-Dunn test. All classifiers whose rank is outside the marked interval are significantly different from the baseline random classifier at $\alpha = 0.05$.

Based on Figure 9.2, the MLP algorithm with tanh activation function performs significantly better than the baseline random classifier with a confidence level of 90% for the Nemenyi test, and with a confidence level of 95% for the Bonferroni-Dunn test (Figure 9.4). Multiple classifiers have average ranks that are in both intervals. The group of those classifiers cannot be determined and it can thus be stated that the experimental data do not allow to find significant differences for those algorithms.

9.4.2 Comparison of the classifiers for the repetition number prediction task

Starting from this section, each of the 200 resamples is considered as a different viewpoint of the data set and the models are compared for each prediction task, for each resample. The 200 balanced accuracy scores and the corresponding ranks that led to this representation are given in Table A.1. The computation of the Friedman test and the improved Friedman statistic can be found in Equation 9.5 and Equation 9.6. Since there are 7 algorithms and 200 data sets, F_F is distributed according to the F distribution with $200 - 1 = 199$ and $(200 - 1) \times (7 - 1) = 1194$ degrees of freedom. The critical value of $F(199, 1194)$ for $\alpha = 0.05$ is 1.19. These results show that the null hypothesis, i.e. all the classifiers have equal results, can be rejected as $1.19 < 62.9$.

$$\begin{aligned}\chi_F^2 &= \frac{12 \cdot 200}{7 \cdot 8} \left[(6.122^2 + 4.54^2 + 3.69^2 + 2.852^2 + 3.678^2 + 3.682^2 + 3.435^2) - \frac{7 \cdot 8^2}{4} \right] \\ &= 288.19\end{aligned}\tag{9.5}$$

$$F_F = \frac{(200 - 1) \cdot 288.19}{200(7 - 1) - 288.19} = 62.9\tag{9.6}$$

A graphic comparison of the classifiers for the Nemenyi test is shown on figure 9.5. The corresponding CD for $\alpha = 0.05$ is calculated in the same way as 9.4.1 and the value of 1.56 is obtained. By the sight of this figure, it can be stated that all classifiers are statistically better than the baseline random classifier. Then, the MLP+r, MLP, MLP+t and SVM+l classifiers are not significantly different from each other, but significantly better than the SVM and the baseline random classifier. It is also showed that for this repetition number prediction task, the tuning of the SVM hyperparameters statistically improves the performances, since

the performances of SVM+l and SVM+r models are significantly different of those of the SVM model, but the same cannot be proven for the MLP models.

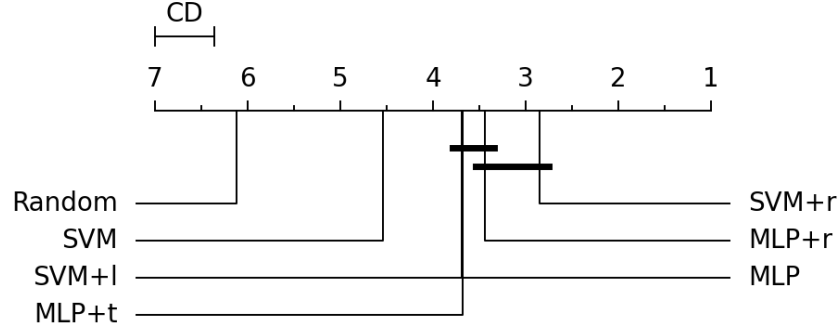


Figure 9.5: Comparison of all classifiers against each other using the Nemenyi test for the repetition number prediction task. Classifiers belonging to the same group (connected with a black line) do not have performances significantly different at $\alpha = 0.05$.

9.4.3 Comparison of the classifiers for the age range prediction task

Normal age data set

As for the repetition number prediction task, there are 7 algorithms and 200 data sets, which gives the same critical value of 1.19 for $\alpha = 0.05$. Comparing the result of Equation 9.8 with this critical value shows that the null hypothesis can be rejected as well for the age range prediction task as $1.19 < 108.32$.

$$\begin{aligned}\chi_F^2 &= \frac{12 \cdot 200}{7 \cdot 8} \left[(6.442^2 + 4.872^2 + 3.12^2 + 3.17^2 + 4.06^2 + 2.78^2 + 3.555^2) - \frac{7 \cdot 8^2}{4} \right] \\ &= 422.96\end{aligned}\tag{9.7}$$

$$F_F = \frac{(200 - 1) \cdot 422.96}{200(7 - 1) - 422.96} = 108.32\tag{9.8}$$

The graphic comparison of the classifiers for the Nemenyi post-hoc test is shown on figure 9.6. As for the previous subsection, the 200 balanced accuracy scores and the corresponding ranks that led this representation are given in Table A.2. The corresponding CD for $\alpha = 0.05$ is 1.56. All the classifiers perform significantly

better than the baseline random algorithm. The best ranked model is MLP+t. Its performances are not significantly different from SVM+l and SVM+r, but significantly better than all the other algorithms. Moreover, it can be stated that the tuning of the hyperparameters of the SVM statistically improves the performances, and the use of the tanh activation function for the MLP models statistically improves it as well.

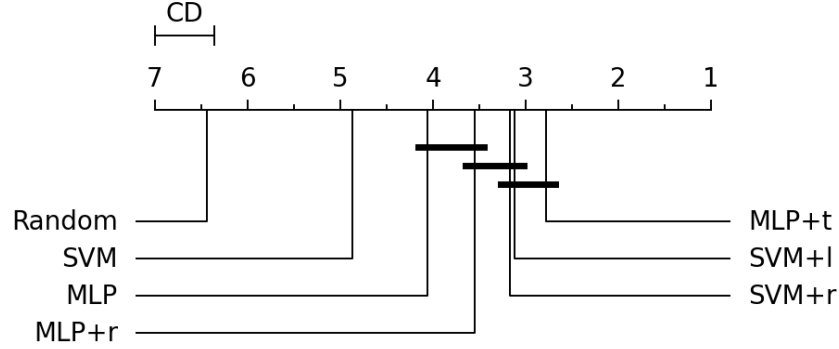


Figure 9.6: Comparison of all classifiers against each other using the Nemenyi test for the age range prediction task. Classifiers belonging to the same group (connected with a black line) do not have performances significantly different at $\alpha = 0.05$.

Denoised age data set

The critical value of 1.19 for $\alpha = 0.05$ is the same as presented earlier as the conditions of the Friedman test haven't changed. Comparing the result of Equation 9.8 with this critical value shows that the null hypothesis can also be rejected as $1.19 < 74.2$ for the prediction task of the denoised age data set.

$$\begin{aligned}\chi_F^2 &= \frac{12 \cdot 200}{7 \cdot 8} \left[(5.722^2 + 5.368^2 + 3.955^2 + 3.1785^2 + 3.09^2 + 3.195^2 + 2.885^2) - \frac{7 \cdot 8^2}{4} \right] \\ &= 325.9\end{aligned}\tag{9.9}$$

$$F_F = \frac{(200 - 1) \cdot 325.9}{200(7 - 1) - 325.9} = 74.2\tag{9.10}$$

The graphic comparison of the classifiers for the Nemenyi post-hoc test is shown on figure 9.7. The 200 balanced accuracy scores and the corresponding ranks

that led to this representation are given in Table A.3. The corresponding CD for $\alpha = 0.05$ is 1.56. Surprisingly, the three best-ranked classifiers are not exactly the same than for the normal age data set. Here, the three best-ranked classifiers are the three versions of MLP. They are followed by the three SVM versions and finally the random baseline classifier. All classifiers perform significantly better than these baseline random classifier except the basic SVM. The two tuned SVM have performances significantly higher than the basic SVM, but the same cannot be proven for the MLP models.

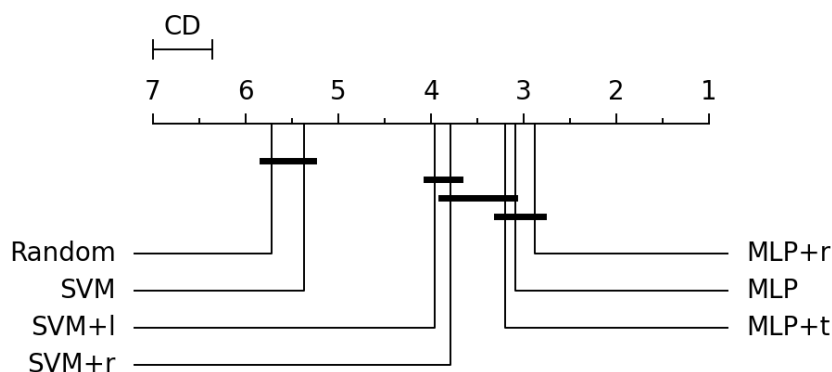


Figure 9.7: Comparison of all classifiers against each other using the Nemenyi test for the age range prediction task and with the denoised data set. Classifiers belonging to the same group (connected with a black line) do not have performances significantly different at $\alpha = 0.05$.

9.5 Denoised data for the age range prediction

In order to evaluate whether the denoised version of the data set improves the age range prediction, a paired t-test between the overall best averaged ranked classifier (MLP+t) with normal data and this same classifier with denoised data was performed by taking the 200 balanced accuracy results of this classifier for these two data sets. The two-sided obtained p-value was 1.96×10^{-7} . Thus, for the Multi-Layer Perceptron with the activation function tanh, the hypothesis of identical performances can be rejected. The denoised data gives statistically better results.

9.6 Discussion

Results significance. Depending on the choice of design, and more particularly on what is considered a data set, we may or may not obtain significant differences between the performances of the different algorithms. By comparing the classifiers as in section 9.4.1, the comparison is made for the type of data used but independently of the prediction task. However, as we see in the following sections (sections 9.4.2 and 9.4.3), the most appropriate classifier for age range prediction is not necessarily the same as for repetition number prediction. Moreover, the comparison in section 9.4.1 is based on only three data sets, whereas the comparison of classifiers for each specific task (sections 9.4.2 and 9.4.3) is based on what is considered to be 200 data set viewpoints. Therefore, the 200 performance measures could have highlighted the previous trend of the classifiers, computed on only 3 data sets, which could have been due to chance.

Training effect. As expected, the results for the prediction of the repetition are quite low. This forces to consider whether the training effect has a small influence on these movement data or if the data set doesn't reflect enough of this training effect through its features. The first case could be plausible as repeating the same task only three times in a row may not result in big differences. Moreover, the difference could be more notable between the first and the third repetition than between two consecutive repetitions. The latter could indicate that the features that are analysed for the four selected tasks, FT, PS, PT and KT, are not very sensitive to a training effect.

Age effect. The performances for the age range prediction are also quite low (averaged balanced accuracy on the 200 resamples for the MLP+t model of 46.96% for the normal data set and of 52.74% for the denoised data set). These low performances could be explained by several factors. First, as described in chapter 6, the data contains outliers. These outliers are scaled before being used to compute PCA components but could still have an influence on the training process of the models. Another factor to take into account is that the samples with class 2, representing the ages ranging from 65 to 87, are very often miss-classified. As a reminder, this class is only represented by 5 volunteers. These 5 volunteers were very healthy and sportive. Unfortunately, this may not represent the population of this age group very well. Moreover, since both the age groups 1 and 2 are underrepresented compared to age group 0, this may cause problems as the classifier is not able to generalize over the samples of the minority classes during training.

Best classifier. The best classifier depends on the prediction task. MLP+t is the best overall ranked classifier. However, according to the Nemenyi post-hoc test with $\alpha = 0.05$, the SVM+r model is significantly better if we only consider the repetition number prediction task. Moreover, for the age range prediction task, MLP+t is the best ranked model for the normal age data sets, and MLP+r is best ranked for the denoised dataset. Its performances are however not significantly different from MLP+t model, according to the Nemenyi post-hoc test with $\alpha = 0.05$. Let us add that for the comparison of the classifiers with the resamples of the denoised data set, the three versions of SVM model have lower ranks than the three versions of the MLP model. One possible problem with the SVM could be the curse of dimensionality: with a high dimensional feature space (higher than the original with a rbf kernel or of the same size with a linear kernel) and only 59 samples, the data set becomes sparse and may not be enough to separate the data correctly. Moreover, SVM works by finding a boundary (linear or not) between each class. However, maybe more than one boundary could be needed to separate each pair of classes for the age range prediction. This could be a reason for the poorer results.

Parameters of MLP Neural networks are very powerful tools given that they have the right hyperparameters. Analysing the choice of some of these hyperparameters in the hyperparameter tuning for 200 resamples may help determining how these models could be improved. For the alpha value, the value 0.1 was chosen 169 times, 0.01 was chosen 30 times and 0.001 only once for MLP+t. Approximately the same proportions for the choice of alpha are found in the results of MLP+r. This indicates that for predictions on the age data set, the classifiers tend to choose a higher penalty parameter than the default value. As a reminder, this value contributes to keeping the weights of the neural network low in order to avoid over-fitting, particularly in this case where only a limited set of training samples are available or in the case where noise is present in the data. Higher values might indicate that the model performs better when over-fitting is heavily penalized. The choice of the activation function can also be analysed. Unlike what could be seen in section 7.3, the MLP with a ReLu activation function does not appear to perform better than a MLP with the hyperbolic tangent activation function. For the comparison of the classifiers with the resamples of the normal age data set, it is even the other way around : the MLP model with a hyperbolic tangent activation function is significantly better than the one with the ReLu activation function (according to the Nemenyi test with $\alpha = 0.05$). The difference is however not significant for the comparison with the denoised age data and the one with the repetition number data set.

Parameters of SVM According to the Nemenyi tests computed for the repetition number and the age range prediction tasks, the tuning of hyperparameters significantly improve the models: SVM+l and SVM+r have significantly better performances than SVM. Moreover, a surprising fact can be noted : unlike the analysis made in section 6.3 about the absence of linearity in the data and the results obtained in this chapter, the SVM with a rbf (non-linear) kernel gives significant better performances only for the repetition number prediction task. With the normal and denoised age data sets, the difference of performance is not significant.

The denoised age. This data set seems to allow for higher performances in general : for all classifiers except one (default SVM) the averaged balanced accuracy score of the 200 resamples is higher for the denoised age data set than for the normal age data set. Moreover, on the best averaged ranked classifier, MLP with the activation function tanh, the denoised data gives statistically a better balanced accuracy score. This may confirm the hypothesis of section 6.4 where the noise effect that could be present in the multiple samples per volunteer and the possible way to reduce it is described. This denoised effect seems however to lead to more improved performances on the MLP classifiers than for the SVM ones. The fact that this technique seems to improve the performances on several classifiers could be explained by two factors: machine-initiated noise and volunteer-initiated noise. The latter comes from the fact that a volunteer does not perform the same movement twice in exactly the same way. Thus, having a volunteer performing the same task multiple times can help to decrease noise by taking the median value of the records and under the assumption that the training effect is negligible.

Dimensionality reduction. For our tests, only one dimensionality reduction technique has been used. However, PCA is not based on removing features but on computing linear combinations of the existing ones to replace them. Using this technique, we have also kept as much variance of the original data set as possible given its small size. Since 99% of the variance of the original data set has been kept, some of the noise from the features is still present and could have an influence on the performances of our models. Nevertheless, both algorithms that are presented, SVM and MLP, have what are called regularization parameters. For SVM classifiers, this allows them to penalise miss-classifications whereas for the MLP this parameter serves as a penalty term that keeps the learned weights low to avoid overfitting.

Conclusion

This thesis was done in three steps. In the first one, a procedure has been built to collect movement data with the SensorMotors, the tool developed by Tools4Patient. The movements are 4 specific tasks of the motor examination part of MDS-UPDRS: FT and PS that focus on speed and dexterity, and PS and KT that are related to tremors. The data from 59 healthy people, i.e. people not suffering from any upper limb movement disorder nor any neurological disorder has then been collected. The SensorMotors recorded the accelerations and the angular velocities of the thumb and the index finger during these tasks. The second step has been to preprocess and extract features from these preprocessed records to build a data set. Finally, two machine learning algorithms have been studied to both make age range and repetition number predictions.

Let us now recall the three main questions of this thesis.

What are the best machine learning approaches to analyze these movement data ? Two families of models were compared and for each, several values of hyper parameters have been tested. By comparing classifiers for the data without regard to a specific prediction task, the MLP with an hyperbolic tangent activation function appears to be the best ranked model. However, considering only one prediction task at a time, the model giving the best performance is not always the same and depends on the prediction task.

What is the influence of the age on the quality of these movements ? Age has showed to be a factor of influence on the data from the four selected tasks of the MDS-UPDRS even though its presence is weak : the balanced accuracy scores of the predictions are quite low, but the models have performances significantly different from those of a baseline random classifier. The effect of the age is thus present in the data and it could bias the data when used, as planned by Tools4Patient, to predict a score for the progression of Parkinson's disease. Moreover, it is important to note that the age of the volunteers who participated in the study is not balanced: half of the participants were 31 years old or younger, only

5 participants, all in very good health were over the age of 60. The performances could therefore perhaps be better with a more balanced representation of the ages.

How is the training effect marked on these movements ? As for the age range predictions, the performances of the repetition number predictions are quite low, and the training effect seems therefore very slight. It is however present in the data, since all models except one have performances significantly better than those of the baseline random classifier.

Thence, we have shown that both age and repetition effect could be noted on the movement data of selected tasks from MDS-UPDRS. They are slightly but still affected.

Let us add that, on the assumption that the training effect would be small, we proposed the idea of summarizing, for each volunteer, all the repetitions of the same task by taking for each feature its median value. The goal was to potentially improve the prediction of the age range and for the MLP with a hyperbolic tangent activation function, it statistically improved the results.

Improvements

While the thesis answered the initial questions, there are still areas for improvement that could be explored in further work. Here are some of them.

Firstly, the feature extraction step has allowed to work with features that are mainly derived from articles that study data given an MDS-UPDRS score prediction context. Nevertheless, other interesting features could still be sought.

Then, as mentioned in section 9.6, dimensionality reduction has allowed to reduce the time and memory needed to train models but this hasn't removed potential noisy features. A feature selection process such as recursive feature selection might have been achieved. Moreover, reducing the number of features through PCA has removed the interpretability of the original features. A selection process could provide more information on the features that best improve predictions.

Moreover, two types of algorithms have been analysed and used in the tests to detect age and training effects on movement data. Expanding this study to other types of algorithms such as nearest-neighbours algorithms or decision tree algorithms could improve the comparison between machine learning algorithms for

features extracted from movement data.

In addition, this thesis faced imbalanced data, which is a very common issue in real-life data sets. In this study, built-in hyperparameters from the tested classifiers have been used to overcome this issue. However, more techniques exist to balance data sets such as majority class under-sampling or minority class over-sampling and even advanced techniques that combine both. They could be explored in further work.

Furthermore, in order to possibly denoise the data, we have made a proposition to summarize the six repetitions (three repetitions per hand) by taking the median of the values. However, other ways to summarize data could be explored, such as computing the mean, or computing the median or the mean of the data but for each hand separately, or as a last option increase the number of repetitions.

Let us add that it was assumed during the study that the effect of age and repetition could be present in all the tasks. However, some tasks might be more affected as people become older or the more they repeat a certain movement. Performing a task-based analysis of age and training effect might highlight this and could also be done in further work.

Finally, this thesis focuses on only four specific tasks of MDS-UPDRS among the 18 covering the motor examination part. Nevertheless, other tasks from this part of the MDS-UPDRS could benefit from our analysis and give a more precise idea of the age and repetition effect on motor tasks. Among the eighteen tasks of the third part of MDS-UPDRS, the analysis performed in this study could be extended to the following tasks: hand movements (3.5), toe tapping (3.7), leg ability (3.8) and rest tremor amplitude (3.17). For all these tasks, the SensorMotors can either be placed on the fingers or the toes of a volunteer and the evaluation criteria of physicians is almost identical to the criteria of the four tasks analysed in this study: tremor, speed, hesitations, halts and amplitude.

Bibliography

- [1] 1.4. *Support Vector Machines — scikit-learn 0.22.2 documentation*. URL: <https://scikit-learn.org/stable/modules/svm.html> (visited on 05/04/2020).
- [2] C. G. Goetz et al. “Movement Disorder Society sponsored revision of the Unified Parkinson’s Disease Rating Scale (MDS-UPDRS): Scale presentation and clinimetric testing results”. In: *Movement Disorders* Vol. 23, no. 15 (2008).
- [3] Adriano O. Andrade et al. “Human Tremor: Origins, Detection and Quantification”. en. In: *Practical Applications in Biomedical Engineering* (Jan. 2013). Publisher: IntechOpen. DOI: 10.5772/54524. URL: <https://www.intechopen.com/books/practical-applications-in-biomedical-engineering/human-tremor-origins-detection-and-quantification> (visited on 04/28/2020).
- [4] J. Benesty. *Estimation spectrale Méthodes non-paramétriques*. fr. Library Catalog: www.yumpu.com. URL: <https://www.yumpu.com/fr/document/read/16852037/estimation-spectrale-methodes-non-parametriques> (visited on 04/26/2020).
- [5] The SciPy community. *scipy.signal.welch — SciPy v1.5.0.dev0+5bb9177 Reference Guide*. URL: <http://scipy.github.io/devdocs/generated/scipy.signal.welch.html> (visited on 04/26/2020).
- [6] Houde Dai, Pengyue Zhang, and Tim C. Lueth. “Quantitative Assessment of Parkinsonian Tremor Based on an Inertial Measurement Unit”. en. In: *Sensors* 15.10 (Oct. 2015). Number: 10 Publisher: Multidisciplinary Digital Publishing Institute, pp. 25055–25071. DOI: 10.3390/s151025055. URL: <https://www.mdpi.com/1424-8220/15/10/25055> (visited on 04/28/2020).
- [7] Janez Demsar. “Statistical Comparisons of Classifiers over Multiple Data Sets”. In: *Journal of Machine Learning Research* 7 (Jan. 2006), pp. 1–30.

- [8] Günther Deuschl, Peter Bain, and Mitchell Brin. “Consensus Statement of the Movement Disorder Society on Tremor”. en. In: *Movement Disorders* 13.S3 (1998), pp. 2–23. ISSN: 1531-8257. DOI: 10.1002/mds.870131303. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1002/mds.870131303> (visited on 04/28/2020).
- [9] O. J. Dunn. “Mutliple comparisons among means”. In: *Journal of the American Statistical Association* 56 (1961), pp. 52–64.
- [10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. <http://www.deeplearningbook.org>. MIT Press, 2016.
- [11] Giuliana Grimaldi and Mario Manto. “Neurological Tremor: Sensors, Signal Processing and Emerging Applications”. en. In: *Sensors* 10.2 (Feb. 2010). Number: 2 Publisher: Molecular Diversity Preservation International, pp. 1399–1422. DOI: 10.3390/s100201399. URL: <https://www.mdpi.com/1424-8220/10/2/1399> (visited on 04/28/2020).
- [12] InvenSense Inc. *MPU-9250 Product Specification Revision 1.1*. 2016.
- [13] Joseph Jankovic, Kenneth S. Schwartz, and William Ondo. “Re-emergent tremor of Parkinson’s disease”. In: *Journal of Neurology, Neurosurgery & Psychiatry* 67.5 (Nov. 1999), pp. 646–650. ISSN: 0022-3050, 1468-330X. DOI: 10.1136/jnnp.67.5.646. URL: <https://jnnp.bmj.com/content/67/5/646>.
- [14] Hyoseon Jeon et al. “Automatic Classification of Tremor Severity in Parkinson’s Disease Using a Wearable Device”. en. In: *Sensors* 17.9 (Sept. 2017). Number: 9 Publisher: Multidisciplinary Digital Publishing Institute, p. 2067. DOI: 10.3390/s17092067. URL: <https://www.mdpi.com/1424-8220/17/9/2067> (visited on 04/28/2020).
- [15] Myeong Hun Jeong, Shaowen Wang, and Clair Sullivan. “Analysis of dynamic radiation level changes using surface networks”. In: Feb. 2016, pp. 199–. ISBN: 9780985244446.
- [16] Donatas Lukšys, Gintaras Jonaitis, and Julius Griškevičius. “Quantitative Analysis of Parkinsonian Tremor in a Clinical Setting Using Inertial Measurement Units”. In: *Parkinson’s Disease* 2018 (June 2018). Ed. by Cristine Alves da Costa. Publisher: Hindawi, p. 1683831. ISSN: 2090-8083. DOI: 10.1155/2018/1683831. URL: <https://doi.org/10.1155/2018/1683831>.
- [17] *MDS-Unified Parkinson’s Disease Rating Scale (MDS-UPDRS)*. URL: <https://www.movementdisorders.org/MDS/MDS-Rating-Scales/MDS-Unified-Parkinsons-Disease-Rating-Scale-MDS-UPDRS.htm> (visited on 04/24/2020).

- [18] Lawrence Mosley. “A balanced approach to the multi-class imbalance problem”. In: *Graduate Theses and Dissertations* (Jan. 2013). DOI: <https://doi.org/10.31274/etd-180810-3375>.
- [19] P. B. Nemenyi. “Distribution-free multiple comparisons”. PhD thesis. 1963.
- [20] *Parkinson’s Disease: Hope Through Research*. Hope Through Research. National Institute of Neurological Disorders and Stroke, National Institutes of Health, Dec. 2014. URL: https://www.ninds.nih.gov/sites/default/files/parkinsons_htr_english_20-ns-139_508c.pdf.
- [21] Ronald M. Ruff and Stephen B. Parker. “Gender- and age-specific changes in motor speed and eye-hand coordination in adults: Normative values for the finger tapping and grooved pegboard tests”. In: *Perceptual and Motor Skills* 76 (1993), pp. 1219–1230.
- [22] Tools 4 Patient SA. *SensorMotor instructions of Use*. Dec. 2017.
- [23] J. Stamatakis et al. “Finger Tapping Clinimetric Score Prediction in Parkinson’s Disease Using Low-Cost Accelerometers”. In: *Computational Intelligence and Neuroscience* Vol. 2013 (2013).
- [24] *Statistics*. en. Library Catalog: www.parkinson.org. URL: <https://www.parkinson.org/Understanding-Parkinsons/Statistics> (visited on 04/24/2020).
- [25] *Support Vector Machines (SVM) / Learn OpenCV*. URL: <https://www.learnopencv.com/support-vector-machines-svm/> (visited on 05/04/2020).
- [26] TF_SensorMotor_T4P. *Technical File SensorMotor*. Feb. 2018.
- [27] Inc. The MathWorks. *Welch’s power spectral density estimate - MATLAB pwelch - MathWorks France*. URL: <https://fr.mathworks.com/help/signal/ref/pwelch.html> (visited on 04/26/2020).
- [28] P. Welch. “The use of the fast Fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms”, *IEEE Trans. Audio Electroacoust.* In: *IEEE Trans. Audio Electroacoust* 15 (1967), pp. 70–73.
- [29] M. Yokoe et al. “Opening velocity, a novel parameter, for finger tapping test in patients with Parkinson’s disease”. In: *Parkinsonism and Related Disorders* 15 (2009), pp. 440–444.

Appendix A

Results of data set resamples

Table A.1: Age prediction results for 200 resamples with ranks.

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.3843 (6)	0.4606 (2)	0.5324 (1)	0.4236 (3)	0.3819 (7)	0.4213 (4)	0.4005 (5)
0.3727 (7)	0.3889 (6)	0.5162 (2)	0.456 (5)	0.4792 (3.5)	0.5463 (1)	0.4792 (3.5)
0.2847 (7)	0.419 (3)	0.4421 (1)	0.3981 (6)	0.4028 (4.5)	0.4352 (2)	0.4028 (4.5)
0.3472 (7)	0.4074 (5)	0.4213 (2)	0.3843 (6)	0.4167 (3.5)	0.4282 (1)	0.4167 (3.5)
0.2523 (7)	0.3866 (6)	0.4491 (5)	0.5625 (1)	0.4977 (3)	0.5116 (2)	0.4931 (4)
0.3472 (7)	0.375 (6)	0.5463 (1)	0.4722 (5)	0.4769 (4)	0.5208 (2)	0.4838 (3)
0.3148 (7)	0.3588 (4.5)	0.4282 (1)	0.412 (2)	0.3588 (4.5)	0.3495 (6)	0.3773 (3)
0.3032 (7)	0.4259 (3)	0.4907 (2)	0.5069 (1)	0.3611 (5.5)	0.419 (4)	0.3611 (5.5)
0.3356 (7)	0.3889 (3)	0.3449 (5)	0.463 (2)	0.3611 (4)	0.5023 (1)	0.3426 (6)
0.3588 (7)	0.419 (5)	0.4977 (2)	0.4884 (3)	0.3935 (6)	0.588 (1)	0.4745 (4)
0.2523 (7)	0.4491 (5)	0.4769 (4)	0.5116 (1)	0.4398 (6)	0.5 (3)	0.5069 (2)
0.3356 (6)	0.3125 (7)	0.5324 (1)	0.5139 (2)	0.4514 (3.5)	0.3819 (5)	0.4514 (3.5)

Continuation of Table A.1

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.2847 (7)	0.4375 (1)	0.3009 (6)	0.3472 (5)	0.3704 (2.5)	0.3565 (4)	0.3704 (2.5)
0.3472 (7)	0.3935 (2)	0.4213 (1)	0.3495 (6)	0.3866 (4)	0.3889 (3)	0.3681 (5)
0.3657 (7)	0.4444 (6)	0.5509 (2)	0.5255 (3)	0.4907 (5)	0.6181 (1)	0.4954 (4)
0.2523 (7)	0.4005 (6)	0.4583 (4)	0.4977 (2)	0.456 (5)	0.5972 (1)	0.4769 (3)
0.4051 (5)	0.4444 (4)	0.456 (3)	0.4606 (2)	0.3542 (6.5)	0.4722 (1)	0.3542 (6.5)
0.3102 (7)	0.4444 (6)	0.5671 (2)	0.5069 (4)	0.5069 (4)	0.7037 (1)	0.5069 (4)
0.3542 (7)	0.4444 (4)	0.6389 (1)	0.4653 (3)	0.4398 (5.5)	0.5347 (2)	0.4398 (5.5)
0.3866 (7)	0.3935 (6)	0.5046 (2)	0.5023 (3)	0.4676 (4.5)	0.5556 (1)	0.4676 (4.5)
0.3102 (7)	0.3866 (6)	0.4005 (4)	0.412 (3)	0.4144 (1.5)	0.3889 (5)	0.4144 (1.5)
0.3148 (7)	0.4144 (6)	0.4745 (3)	0.5023 (2)	0.5139 (1)	0.4352 (5)	0.4514 (4)
0.2847 (7)	0.463 (6)	0.5486 (2)	0.5972 (1)	0.5185 (5)	0.5417 (3)	0.537 (4)
0.3171 (7)	0.3935 (6)	0.5602 (1)	0.4421 (5)	0.544 (2)	0.5417 (3)	0.4514 (4)
0.3843 (7)	0.4306 (5.5)	0.4514 (3)	0.4954 (1.5)	0.4491 (4)	0.4954 (1.5)	0.4306 (5.5)
0.3171 (5)	0.3681 (3)	0.4074 (2)	0.2917 (7)	0.3333 (4)	0.6412 (1)	0.3125 (6)
0.4236 (7)	0.4259 (5.5)	0.4907 (2)	0.4699 (4)	0.4745 (3)	0.6389 (1)	0.4259 (5.5)
0.3356 (7)	0.463 (3)	0.4306 (5)	0.4861 (2)	0.4259 (6)	0.4514 (4)	0.4884 (1)
0.4676 (3)	0.3935 (6)	0.5926 (1)	0.5139 (2)	0.4537 (4)	0.3611 (7)	0.4213 (5)
0.3472 (7)	0.3889 (5)	0.3843 (6)	0.4051 (4)	0.4352 (1)	0.4097 (3)	0.4236 (2)
0.3241 (7)	0.3866 (6)	0.4097 (5)	0.4468 (3.5)	0.4537 (1)	0.4514 (2)	0.4468 (3.5)
0.2708 (7)	0.3333 (6)	0.5949 (1)	0.412 (4)	0.4005 (5)	0.4236 (3)	0.544 (2)

Continuation of Table A.1

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.3426 (7)	0.3681 (6)	0.4282 (1)	0.3958 (4)	0.3912 (5)	0.4028 (3)	0.4167 (2)
0.3495 (7)	0.3704 (6)	0.4491 (5)	0.456 (4)	0.463 (2)	0.4606 (3)	0.4745 (1)
0.3912 (3)	0.3681 (4)	0.4699 (1)	0.3981 (2)	0.3657 (5)	0.3171 (7)	0.3588 (6)
0.3727 (4)	0.4236 (2)	0.3356 (6)	0.4421 (1)	0.3333 (7)	0.4074 (3)	0.3657 (5)
0.3981 (7)	0.412 (4)	0.4005 (6)	0.4213 (3)	0.4097 (5)	0.4653 (1)	0.4259 (2)
0.2986 (7)	0.4259 (1)	0.3056 (6)	0.3796 (4)	0.3912 (2)	0.3333 (5)	0.3843 (3)
0.3102 (7)	0.4375 (2)	0.4051 (4.5)	0.4514 (1)	0.4028 (6)	0.4236 (3)	0.4051 (4.5)
0.2847 (7)	0.3935 (3)	0.3819 (4.5)	0.4282 (2)	0.3565 (6)	0.5718 (1)	0.3819 (4.5)
0.2847 (7)	0.5046 (5)	0.4884 (6)	0.588 (1)	0.5162 (3.5)	0.537 (2)	0.5162 (3.5)
0.3472 (7)	0.4074 (6)	0.4769 (3)	0.4838 (2)	0.463 (4)	0.6111 (1)	0.4514 (5)
0.3565 (7)	0.4444 (5)	0.4005 (6)	0.4514 (4)	0.5417 (2)	0.5694 (1)	0.5208 (3)
0.2593 (7)	0.4375 (5)	0.5903 (1)	0.4005 (6)	0.5231 (3.5)	0.5671 (2)	0.5231 (3.5)
0.4028 (7)	0.4352 (5)	0.4282 (6)	0.5301 (2)	0.4861 (4)	0.544 (1)	0.4931 (3)
0.2523 (7)	0.331 (6)	0.5 (2)	0.4144 (5)	0.4977 (3)	0.4792 (4)	0.5185 (1)
0.3843 (6)	0.3796 (7)	0.5556 (2)	0.456 (4)	0.4375 (5)	0.5764 (1)	0.5116 (3)
0.3403 (7)	0.3843 (6)	0.4028 (4)	0.4144 (3)	0.4514 (2)	0.5324 (1)	0.3935 (5)
0.2778 (7)	0.3519 (6)	0.3912 (3)	0.4398 (1)	0.3611 (5)	0.4282 (2)	0.3727 (4)
0.3102 (7)	0.4491 (5)	0.6273 (1)	0.5394 (3)	0.4838 (4)	0.5926 (2)	0.4352 (6)
0.3843 (7)	0.4259 (5)	0.4491 (3)	0.4653 (2)	0.4097 (6)	0.5046 (1)	0.4375 (4)
0.3333 (7)	0.4491 (3)	0.463 (2)	0.4213 (4)	0.3796 (5.5)	0.4838 (1)	0.3796 (5.5)

Continuation of Table A.1

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.2778 (7)	0.3681 (6)	0.4213 (3)	0.3843 (5)	0.4074 (4)	0.4676 (1)	0.4282 (2)
0.3148 (7)	0.3519 (6)	0.4699 (1)	0.4329 (2)	0.3843 (5)	0.4306 (3)	0.3981 (4)
0.2778 (7)	0.3889 (4)	0.3981 (2)	0.3912 (3)	0.3634 (6)	0.4792 (1)	0.3796 (5)
0.3241 (7)	0.463 (6)	0.5185 (3)	0.5394 (1)	0.5162 (4.5)	0.5324 (2)	0.5162 (4.5)
0.2523 (7)	0.4028 (6)	0.5231 (3)	0.5718 (2)	0.4236 (4.5)	0.6065 (1)	0.4236 (4.5)
0.3912 (2)	0.3681 (5)	0.4375 (1)	0.375 (3)	0.3218 (7)	0.3542 (6)	0.3704 (4)
0.4352 (6)	0.3704 (7)	0.4977 (3)	0.5417 (1)	0.4398 (4.5)	0.5046 (2)	0.4398 (4.5)
0.4421 (4)	0.331 (7)	0.3889 (5.5)	0.3889 (5.5)	0.4468 (3)	0.4884 (1)	0.4769 (2)
0.2963 (7)	0.3819 (6)	0.5023 (3)	0.5116 (2)	0.4977 (4.5)	0.5926 (1)	0.4977 (4.5)
0.3843 (7)	0.3935 (5)	0.3866 (6)	0.5093 (2)	0.4977 (3)	0.4583 (4)	0.5463 (1)
0.3356 (7)	0.463 (5)	0.4954 (3)	0.4398 (6)	0.5301 (1)	0.4699 (4)	0.5162 (2)
0.3032 (7)	0.3519 (6)	0.5208 (1)	0.4745 (2)	0.3981 (5)	0.4653 (3)	0.4306 (4)
0.2847 (7)	0.4375 (3)	0.4537 (2)	0.4699 (1)	0.4051 (5)	0.4005 (6)	0.4167 (4)
0.4097 (6)	0.4537 (1)	0.4306 (3)	0.4167 (5)	0.4213 (4)	0.3935 (7)	0.4352 (2)
0.2894 (7)	0.4051 (2)	0.3611 (5)	0.3542 (6)	0.4028 (3)	0.3958 (4)	0.4167 (1)
0.4352 (7)	0.4444 (6)	0.5949 (2)	0.6366 (1)	0.5 (3)	0.4769 (4)	0.4606 (5)
0.3356 (7)	0.463 (6)	0.4792 (5)	0.5856 (2)	0.5347 (4)	0.6204 (1)	0.5671 (3)
0.3287 (6)	0.3819 (4)	0.4259 (2)	0.3634 (5)	0.3218 (7)	0.4583 (1)	0.4028 (3)
0.2847 (7)	0.3426 (6)	0.412 (3)	0.3958 (5)	0.4051 (4)	0.5949 (1)	0.5 (2)
0.3403 (7)	0.4005 (4)	0.4491 (3)	0.4792 (1)	0.3843 (5.5)	0.4514 (2)	0.3843 (5.5)

Continuation of Table A.1

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.2523 (7)	0.3264 (6)	0.5093 (1)	0.4537 (3)	0.3773 (5)	0.4583 (2)	0.4144 (4)
0.2338 (7)	0.4861 (4)	0.6065 (1)	0.5556 (3)	0.4815 (5)	0.4514 (6)	0.5625 (2)
0.3426 (7)	0.3866 (6)	0.4144 (5)	0.4167 (4)	0.4329 (2.5)	0.456 (1)	0.4329 (2.5)
0.2847 (7)	0.3819 (6)	0.3843 (5)	0.4375 (4)	0.4653 (2)	0.4468 (3)	0.4792 (1)
0.3171 (7)	0.4005 (6)	0.5463 (1.5)	0.5208 (5)	0.5463 (1.5)	0.5417 (3)	0.5347 (4)
0.3333 (6)	0.331 (7)	0.463 (2)	0.3565 (5)	0.4236 (4)	0.4514 (3)	0.4792 (1)
0.3102 (7)	0.3889 (6)	0.4282 (2)	0.4051 (5)	0.412 (4)	0.4815 (1)	0.419 (3)
0.4352 (6)	0.3704 (7)	0.5486 (4)	0.5046 (5)	0.5833 (2)	0.5972 (1)	0.5694 (3)
0.3796 (6)	0.3519 (7)	0.4259 (2)	0.4005 (4)	0.4236 (3)	0.3981 (5)	0.4745 (1)
0.3079 (7)	0.4236 (3)	0.4213 (4)	0.4769 (1)	0.419 (5)	0.4653 (2)	0.4074 (6)
0.3912 (6)	0.3704 (7)	0.6597 (1)	0.4769 (2)	0.4444 (5)	0.4537 (4)	0.4653 (3)
0.3472 (7)	0.3657 (6)	0.5046 (2)	0.4282 (5)	0.4653 (4)	0.4699 (3)	0.5833 (1)
0.3333 (6.5)	0.3333 (6.5)	0.4676 (2)	0.4074 (3)	0.3819 (4.5)	0.4861 (1)	0.3819 (4.5)
0.3287 (6)	0.3681 (2)	0.3194 (7)	0.4028 (1)	0.3449 (5)	0.3472 (4)	0.3519 (3)
0.2917 (7)	0.4491 (6)	0.6134 (2)	0.5509 (3)	0.5069 (4)	0.4861 (5)	0.662 (1)
0.2778 (7)	0.3241 (6)	0.4236 (3)	0.4282 (2)	0.3889 (4.5)	0.5671 (1)	0.3889 (4.5)
0.3403 (7)	0.3681 (5)	0.3935 (3.5)	0.3519 (6)	0.3981 (1)	0.3935 (3.5)	0.3958 (2)
0.4028 (5)	0.4097 (3)	0.4051 (4)	0.3958 (7)	0.4213 (2)	0.3981 (6)	0.4329 (1)
0.2338 (7)	0.3241 (6)	0.3727 (5)	0.3796 (3.5)	0.3935 (1.5)	0.3796 (3.5)	0.3935 (1.5)
0.3843 (7)	0.3935 (6)	0.4954 (1)	0.4861 (2)	0.4306 (4)	0.4421 (3)	0.4282 (5)

Continuation of Table A.1

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.4421 (7)	0.456 (6)	0.5972 (1)	0.5926 (2)	0.5023 (4)	0.5417 (3)	0.4954 (5)
0.2523 (7)	0.3542 (3)	0.3704 (2)	0.3472 (4)	0.3264 (6)	0.4444 (1)	0.3333 (5)
0.4051 (7)	0.463 (5)	0.4537 (6)	0.5417 (3)	0.5162 (4)	0.5602 (1)	0.5486 (2)
0.3356 (6)	0.419 (4)	0.2917 (7)	0.3542 (5)	0.4421 (2)	0.4259 (3)	0.4537 (1)
0.4167 (7)	0.463 (6)	0.588 (2)	0.5463 (3)	0.4722 (5)	0.6019 (1)	0.4977 (4)
0.2593 (7)	0.4444 (6)	0.4722 (4.5)	0.5116 (2)	0.4722 (4.5)	0.6019 (1)	0.5046 (3)
0.2778 (7)	0.4213 (5)	0.4282 (4)	0.3981 (6)	0.4606 (3)	0.5208 (1)	0.4884 (2)
0.3102 (7)	0.3125 (6)	0.3958 (2)	0.4028 (1)	0.338 (4)	0.3704 (3)	0.331 (5)
0.3218 (7)	0.4259 (6)	0.4722 (2)	0.4537 (3.5)	0.4815 (1)	0.4444 (5)	0.4537 (3.5)
0.3356 (7)	0.4491 (5)	0.4144 (6)	0.5023 (2)	0.5185 (1)	0.463 (3)	0.4583 (4)
0.2269 (7)	0.4306 (6)	0.463 (5)	0.4722 (4)	0.5833 (3)	0.6644 (1)	0.6065 (2)
0.4097 (1)	0.3611 (3)	0.3333 (4)	0.3657 (2)	0.3102 (6.5)	0.3194 (5)	0.3102 (6.5)
0.3981 (7)	0.4676 (4)	0.4699 (2.5)	0.4699 (2.5)	0.4606 (5)	0.5532 (1)	0.4259 (6)
0.2454 (7)	0.456 (6)	0.5231 (2)	0.4977 (3)	0.5255 (1)	0.4653 (5)	0.4954 (4)
0.2593 (7)	0.4745 (4)	0.4745 (4)	0.6019 (1)	0.4468 (6)	0.4745 (4)	0.5648 (2)
0.2523 (7)	0.3912 (5)	0.3843 (6)	0.4745 (2)	0.4491 (4)	0.4583 (3)	0.4815 (1)
0.3241 (6)	0.375 (5)	0.412 (2.5)	0.2338 (7)	0.4236 (1)	0.3889 (4)	0.412 (2.5)
0.3912 (2)	0.3866 (3)	0.3403 (6)	0.3819 (4)	0.4421 (1)	0.3356 (7)	0.3657 (5)
0.3171 (7)	0.4421 (4)	0.4653 (3)	0.4769 (1)	0.4282 (6)	0.4745 (2)	0.4375 (5)
0.4352 (3)	0.338 (7)	0.4769 (1)	0.4606 (2)	0.3681 (5.5)	0.4282 (4)	0.3681 (5.5)

Continuation of Table A.1

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.3032 (7)	0.3866 (6)	0.4306 (5)	0.4468 (4)	0.5231 (2.5)	0.5417 (1)	0.5231 (2.5)
0.2847 (7)	0.463 (4)	0.5116 (2)	0.4144 (5)	0.4097 (6)	0.5972 (1)	0.4977 (3)
0.3241 (7)	0.3889 (6)	0.4259 (3)	0.4005 (4)	0.4375 (2)	0.3981 (5)	0.4444 (1)
0.2639 (7)	0.5116 (4)	0.537 (2)	0.5509 (1)	0.463 (6)	0.5324 (3)	0.4977 (5)
0.3102 (3)	0.2986 (5)	0.3287 (2)	0.2569 (7)	0.3009 (4)	0.3472 (1)	0.2963 (6)
0.3102 (7)	0.3935 (6)	0.4653 (4)	0.4931 (1)	0.4769 (2)	0.4676 (3)	0.4444 (5)
0.3796 (7)	0.4259 (5)	0.537 (1)	0.456 (4)	0.4676 (3)	0.4213 (6)	0.4861 (2)
0.2894 (7)	0.3935 (3)	0.3958 (2)	0.3472 (6)	0.3773 (5)	0.4491 (1)	0.3912 (4)
0.3727 (7)	0.3889 (6)	0.4421 (5)	0.5 (4)	0.5162 (2)	0.5602 (1)	0.5046 (3)
0.3148 (7)	0.3796 (6)	0.4884 (1)	0.3958 (4)	0.3935 (5)	0.4861 (2)	0.463 (3)
0.3032 (7)	0.4005 (4)	0.4144 (2)	0.412 (3)	0.3912 (6)	0.4167 (1)	0.3981 (5)
0.2847 (7)	0.338 (6)	0.5069 (1)	0.4699 (3)	0.412 (5)	0.5046 (2)	0.4421 (4)
0.2593 (7)	0.4606 (4)	0.4861 (3)	0.4884 (2)	0.4005 (5)	0.5417 (1)	0.3819 (6)
0.412 (6)	0.4815 (2)	0.4583 (3)	0.4398 (4)	0.4028 (7)	0.5 (1)	0.4236 (5)
0.4606 (7)	0.4931 (6)	0.5972 (2)	0.6157 (1)	0.5833 (3.5)	0.5185 (5)	0.5833 (3.5)
0.3148 (7)	0.3819 (6)	0.4097 (2)	0.3889 (5)	0.3958 (4)	0.4074 (3)	0.4213 (1)
0.3472 (7)	0.4074 (4)	0.4745 (1)	0.456 (2)	0.3843 (6)	0.4329 (3)	0.3981 (5)
0.2917 (7)	0.5 (4)	0.6065 (1)	0.5764 (3)	0.4792 (6)	0.5995 (2)	0.4977 (5)
0.4097 (6)	0.3681 (7)	0.5255 (2)	0.4653 (3.5)	0.4653 (3.5)	0.456 (5)	0.5556 (1)
0.3403 (7)	0.3727 (2)	0.3588 (4)	0.3611 (3)	0.3426 (5.5)	0.4398 (1)	0.3426 (5.5)

Continuation of Table A.1

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.2847 (7)	0.3935 (6)	0.4282 (4)	0.4074 (5)	0.5255 (1.5)	0.4653 (3)	0.5255 (1.5)
0.3912 (6)	0.3889 (7)	0.3935 (5)	0.4028 (4)	0.4398 (2)	0.456 (1)	0.4352 (3)
0.2778 (7)	0.3704 (6)	0.5394 (1)	0.4676 (2.5)	0.419 (4)	0.4676 (2.5)	0.3935 (5)
0.3079 (7)	0.4375 (3.5)	0.4491 (2)	0.4329 (5)	0.4375 (3.5)	0.4167 (6)	0.4514 (1)
0.3542 (7)	0.4444 (4)	0.4653 (2)	0.4792 (1)	0.4306 (5)	0.4005 (6)	0.4583 (3)
0.4676 (4)	0.4005 (7)	0.5579 (1)	0.4792 (3)	0.4606 (5.5)	0.5069 (2)	0.4606 (5.5)
0.4421 (4)	0.419 (6)	0.2917 (7)	0.456 (3)	0.4375 (5)	0.5347 (1)	0.4676 (2)
0.2593 (7)	0.4259 (1)	0.375 (5)	0.3634 (6)	0.412 (3)	0.4167 (2)	0.4074 (4)
0.3542 (6.5)	0.4491 (3)	0.3542 (6.5)	0.4352 (4)	0.4954 (1)	0.4329 (5)	0.4884 (2)
0.2593 (7)	0.463 (2.5)	0.4329 (4)	0.463 (2.5)	0.4236 (5.5)	0.4699 (1)	0.4236 (5.5)
0.2523 (7)	0.3796 (6)	0.4977 (2)	0.4491 (4)	0.4792 (3)	0.4375 (5)	0.5231 (1)
0.3426 (5)	0.3333 (7)	0.3356 (6)	0.3611 (3)	0.3588 (4)	0.3773 (1)	0.3681 (2)
0.3171 (7)	0.4051 (4)	0.331 (6)	0.3819 (5)	0.4375 (2)	0.4097 (3)	0.463 (1)
0.4352 (1.5)	0.412 (4)	0.4352 (1.5)	0.4259 (3)	0.338 (6)	0.3356 (7)	0.3519 (5)
0.3079 (4)	0.3819 (1.5)	0.2847 (5.5)	0.3171 (3)	0.2824 (7)	0.3819 (1.5)	0.2847 (5.5)
0.3079 (7)	0.4074 (6)	0.4352 (5)	0.4444 (4)	0.4653 (2)	0.4514 (3)	0.4977 (1)
0.3102 (7)	0.4306 (4)	0.5301 (1)	0.4329 (2.5)	0.4028 (5.5)	0.4329 (2.5)	0.4028 (5.5)
0.2847 (7)	0.3935 (4)	0.3727 (5)	0.4144 (2)	0.419 (1)	0.3958 (3)	0.3542 (6)
0.2778 (7)	0.3889 (6)	0.5625 (1)	0.4838 (4)	0.4861 (3)	0.5255 (2)	0.4167 (5)
0.2639 (7)	0.4074 (6)	0.4236 (5)	0.4977 (2)	0.4282 (3.5)	0.5093 (1)	0.4282 (3.5)

Continuation of Table A.1

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.3171 (7)	0.4074 (5)	0.5972 (1)	0.4977 (2)	0.4051 (6)	0.4699 (3)	0.4398 (4)
0.2847 (7)	0.4421 (5)	0.4583 (3.5)	0.5185 (1)	0.4583 (3.5)	0.4954 (2)	0.419 (6)
0.3912 (7)	0.463 (6)	0.5185 (2)	0.5255 (1)	0.5093 (3)	0.4815 (5)	0.5023 (4)
0.3032 (7)	0.3194 (6)	0.3657 (4)	0.3681 (3)	0.3495 (5)	0.4583 (1)	0.375 (2)
0.3981 (2)	0.3819 (4.5)	0.4167 (1)	0.3866 (3)	0.3472 (6.5)	0.3819 (4.5)	0.3472 (6.5)
0.3356 (7)	0.4676 (6)	0.5139 (2)	0.5394 (1)	0.4954 (4)	0.4722 (5)	0.5023 (3)
0.4421 (3)	0.3819 (6)	0.5231 (1)	0.4097 (4)	0.3519 (7)	0.3981 (5)	0.5093 (2)
0.3287 (7)	0.4259 (6)	0.5046 (1)	0.4977 (2)	0.4468 (5)	0.4838 (3)	0.4722 (4)
0.2662 (5)	0.2986 (1)	0.2801 (3.5)	0.2616 (7)	0.287 (2)	0.2639 (6)	0.2801 (3.5)
0.3542 (6)	0.3333 (7)	0.3981 (4)	0.4167 (3)	0.3611 (5)	0.5532 (1)	0.4236 (2)
0.3102 (7)	0.3935 (6)	0.5255 (5)	0.5486 (3)	0.5648 (1)	0.5301 (4)	0.5509 (2)
0.3542 (7)	0.3634 (6)	0.3843 (4)	0.3912 (3)	0.3773 (5)	0.4329 (1)	0.4028 (2)
0.4097 (4)	0.3889 (7)	0.5509 (2)	0.5556 (1)	0.3935 (5.5)	0.4884 (3)	0.3935 (5.5)
0.3727 (6)	0.3796 (5)	0.3588 (7)	0.4259 (4)	0.463 (1)	0.4606 (2)	0.4375 (3)
0.2894 (7)	0.4306 (4)	0.4931 (1)	0.4375 (3)	0.4167 (5)	0.463 (2)	0.4074 (6)
0.3032 (7)	0.456 (4)	0.5093 (2)	0.5324 (1)	0.3657 (6)	0.4907 (3)	0.412 (5)
0.2269 (7)	0.2986 (5)	0.3519 (1.5)	0.3519 (1.5)	0.2801 (6)	0.3032 (4)	0.3171 (3)
0.3102 (7)	0.4745 (3)	0.412 (6)	0.5046 (1)	0.5023 (2)	0.456 (4)	0.4352 (5)
0.3287 (7)	0.4838 (1)	0.412 (5)	0.4329 (3)	0.3681 (6)	0.4606 (2)	0.4259 (4)
0.3657 (3)	0.3333 (4)	0.3287 (5)	0.2847 (7)	0.3773 (2)	0.3819 (1)	0.3194 (6)

Continuation of Table A.1

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.3403 (7)	0.3889 (4)	0.4722 (1)	0.3981 (3)	0.3866 (5.5)	0.3866 (5.5)	0.456 (2)
0.3588 (7)	0.3935 (6)	0.5972 (3)	0.5648 (4)	0.5463 (5)	0.6806 (1)	0.625 (2)
0.2917 (7)	0.463 (3)	0.4468 (4)	0.4792 (2)	0.4329 (6)	0.5301 (1)	0.4398 (5)
0.4421 (1)	0.3194 (7)	0.4398 (2)	0.4306 (3)	0.3472 (5)	0.3935 (4)	0.3426 (6)
0.3403 (7)	0.3889 (6)	0.5093 (1)	0.4282 (5)	0.4352 (4)	0.4769 (3)	0.4861 (2)
0.3009 (7)	0.338 (6)	0.4653 (3)	0.4468 (5)	0.4907 (2)	0.5648 (1)	0.4583 (4)
0.3102 (7)	0.4444 (4)	0.5463 (1)	0.5093 (2)	0.3935 (6)	0.4954 (3)	0.4074 (5)
0.2778 (7)	0.4074 (4)	0.3403 (6)	0.4699 (1)	0.4491 (2)	0.4468 (3)	0.3912 (5)
0.4375 (7)	0.463 (6)	0.5347 (1)	0.5185 (2)	0.4722 (5)	0.5023 (4)	0.5046 (3)
0.3403 (7)	0.4005 (5)	0.3958 (6)	0.4468 (3.5)	0.4468 (3.5)	0.4745 (2)	0.5394 (1)
0.3796 (7)	0.4236 (4)	0.4861 (1)	0.4306 (3)	0.3981 (6)	0.4352 (2)	0.412 (5)
0.4097 (6)	0.4491 (5)	0.3773 (7)	0.4954 (3)	0.4907 (4)	0.4977 (2)	0.5069 (1)
0.4352 (6)	0.463 (4)	0.412 (7)	0.4722 (3)	0.4838 (1.5)	0.4444 (5)	0.4838 (1.5)
0.2523 (7)	0.4167 (2)	0.5208 (1)	0.4144 (3)	0.3704 (5)	0.4028 (4)	0.3588 (6)
0.2917 (7)	0.463 (4)	0.5417 (1)	0.4745 (3)	0.3935 (6)	0.5301 (2)	0.419 (5)
0.3403 (4)	0.3125 (6)	0.3218 (5)	0.3032 (7)	0.3588 (2)	0.3634 (1)	0.3519 (3)
0.2269 (7)	0.4745 (4)	0.4977 (3)	0.544 (1)	0.4398 (5.5)	0.5069 (2)	0.4398 (5.5)
0.3102 (5)	0.3125 (3)	0.2778 (7)	0.3009 (6)	0.3125 (3)	0.3426 (1)	0.3125 (3)
0.2894 (7)	0.4444 (5)	0.4815 (4)	0.5 (1)	0.4861 (3)	0.4329 (6)	0.4931 (2)
0.3171 (7)	0.3542 (1)	0.3426 (4)	0.3495 (2.5)	0.331 (5)	0.3218 (6)	0.3495 (2.5)

Continuation of Table A.1						
Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.2847 (7)	0.4306 (5)	0.5926 (1)	0.4676 (4)	0.4213 (6)	0.5 (3)	0.5023 (2)
0.3588 (6)	0.3102 (7)	0.4051 (4)	0.3935 (5)	0.456 (2)	0.4398 (3)	0.463 (1)
0.3287 (7)	0.4097 (4)	0.4051 (5)	0.4861 (1)	0.4028 (6)	0.4329 (3)	0.4398 (2)
0.3241 (7)	0.3866 (2.5)	0.3495 (6)	0.3866 (2.5)	0.3773 (4)	0.3588 (5)	0.3912 (1)
0.2338 (7)	0.456 (3)	0.4815 (1)	0.4421 (4)	0.4352 (5.5)	0.4792 (2)	0.4352 (5.5)
0.4028 (7)	0.4398 (6)	0.5023 (1)	0.4491 (4)	0.4769 (2)	0.4444 (5)	0.4653 (3)
0.3171 (7)	0.3727 (6)	0.4815 (2)	0.4931 (1)	0.3866 (4.5)	0.4167 (3)	0.3866 (4.5)
0.2662 (7)	0.4491 (6)	0.4653 (5)	0.5324 (1)	0.4884 (3)	0.4931 (2)	0.4769 (4)
6.442	4.872	3.12	3.17	4.06	2.78	3.555

Table A.2: Age denoised prediction results for 200 resamples with ranks.

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.2083 (7)	0.3333 (5.5)	0.3333 (5.5)	0.4028 (3)	0.4444 (1.5)	0.3889 (4)	0.4444 (1.5)
0.3194 (7)	0.3333 (5.5)	0.3333 (5.5)	0.3611 (4)	0.4028 (2)	0.4028 (2)	0.4028 (2)
0.1667 (7)	0.4444 (2)	0.3472 (5)	0.2361 (6)	0.3611 (3.5)	0.4722 (1)	0.3611 (3.5)
0.3194 (7)	0.4444 (5)	0.3333 (6)	0.4722 (4)	0.5139 (2)	0.5139 (2)	0.5139 (2)
0.3194 (6)	0.4444 (3)	0.2917 (7)	0.5833 (2)	0.4028 (4.5)	0.625 (1)	0.4028 (4.5)
0.4722 (4)	0.3333 (7)	0.4028 (5.5)	0.5556 (2.5)	0.4028 (5.5)	0.625 (1)	0.5556 (2.5)
0.1667 (7)	0.3333 (5)	0.3194 (6)	0.4028 (3)	0.5139 (1)	0.3889 (4)	0.4306 (2)
0.3194 (7)	0.4444 (3)	0.6944 (1)	0.3611 (6)	0.4306 (4.5)	0.5417 (2)	0.4306 (4.5)
0.1667 (7)	0.4444 (2.5)	0.4028 (5)	0.4444 (2.5)	0.4444 (2.5)	0.3194 (6)	0.4444 (2.5)

Continuation of Table A.2

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.4722 (2.5)	0.4444 (5)	0.25 (7)	0.4722 (2.5)	0.4722 (2.5)	0.3889 (6)	0.4722 (2.5)
0.3194 (6.5)	0.4444 (2)	0.3611 (4)	0.3333 (5)	0.4444 (2)	0.3194 (6.5)	0.4444 (2)
0.1667 (7)	0.3333 (6)	0.7778 (1)	0.4444 (5)	0.7361 (3)	0.7361 (3)	0.7361 (3)
0.3194 (7)	0.3333 (5.5)	0.3333 (5.5)	0.5139 (2)	0.5139 (2)	0.4583 (4)	0.5139 (2)
0.3194 (7)	0.4444 (4)	0.4028 (6)	0.5139 (1)	0.4722 (2.5)	0.4306 (5)	0.4722 (2.5)
0.3611 (5)	0.3333 (6.5)	0.5139 (4)	0.3333 (6.5)	0.6667 (1.5)	0.625 (3)	0.6667 (1.5)
0.3194 (6.5)	0.4444 (4)	0.3194 (6.5)	0.5139 (2)	0.5139 (2)	0.4306 (5)	0.5139 (2)
0.3194 (7)	0.4444 (4)	0.3333 (5.5)	0.3333 (5.5)	0.5833 (2)	0.5833 (2)	0.5833 (2)
0.3194 (7)	0.3333 (5.5)	0.4028 (3.5)	0.3333 (5.5)	0.6667 (1)	0.4028 (3.5)	0.5556 (2)
0.3194 (7)	0.4444 (4)	0.3333 (6)	0.5417 (1)	0.4444 (4)	0.5139 (2)	0.4444 (4)
0.3194 (6)	0.3333 (5)	0.2917 (7)	0.4444 (1.5)	0.3611 (4)	0.4306 (3)	0.4444 (1.5)
0.3194 (7)	0.3333 (6)	0.4444 (4)	0.5139 (2)	0.4028 (5)	0.5139 (2)	0.5139 (2)
0.1667 (7)	0.2917 (6)	0.3889 (3.5)	0.3611 (5)	0.5139 (2)	0.5417 (1)	0.3889 (3.5)
0.5417 (4)	0.4444 (5)	0.8889 (1)	0.5556 (3)	0.2917 (6.5)	0.8056 (2)	0.2917 (6.5)
0.1667 (7)	0.3333 (6)	0.8056 (1)	0.5833 (3)	0.4722 (5)	0.5833 (3)	0.5833 (3)
0.3611 (5)	0.2917 (6.5)	0.2917 (6.5)	0.625 (1)	0.4722 (4)	0.5417 (2)	0.5139 (3)
0.3194 (3)	0.3333 (2)	0.25 (5.5)	0.2083 (7)	0.25 (5.5)	0.5 (1)	0.2917 (4)
0.2083 (7)	0.3333 (6)	0.8472 (1.5)	0.5139 (3.5)	0.5139 (3.5)	0.8472 (1.5)	0.4722 (5)
0.3611 (7)	0.4444 (5)	0.4444 (5)	0.5139 (3)	0.5556 (1.5)	0.4444 (5)	0.5556 (1.5)
0.6944 (3)	0.3333 (7)	0.6111 (6)	0.6667 (4)	0.9583 (1.5)	0.6528 (5)	0.9583 (1.5)

Continuation of Table A.2

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.3611 (5)	0.3333 (6.5)	0.3333 (6.5)	0.5556 (2)	0.5556 (2)	0.5139 (4)	0.5556 (2)
0.3194 (7)	0.3333 (6)	0.4722 (4)	0.4028 (5)	0.5139 (1.5)	0.5 (3)	0.5139 (1.5)
0.3194 (7)	0.3333 (6)	0.5139 (1)	0.4722 (2.5)	0.3611 (5)	0.4306 (4)	0.4722 (2.5)
0.6944 (1)	0.4444 (7)	0.5139 (4)	0.5139 (4)	0.5556 (2)	0.4722 (6)	0.5139 (4)
0.3611 (6)	0.3333 (7)	0.4444 (2)	0.4444 (2)	0.4028 (4.5)	0.4028 (4.5)	0.4444 (2)
0.4722 (1)	0.2917 (6.5)	0.2917 (6.5)	0.4028 (3)	0.4028 (3)	0.3611 (5)	0.4028 (3)
0.3194 (5)	0.3333 (4)	0.2917 (6.5)	0.2917 (6.5)	0.5833 (2)	0.5833 (2)	0.5833 (2)
0.4722 (5.5)	0.4028 (7)	0.5833 (2.5)	0.5833 (2.5)	0.4722 (5.5)	0.5417 (4)	0.625 (1)
0.3611 (6)	0.3333 (7)	0.4028 (5)	0.5139 (2)	0.5139 (2)	0.4722 (4)	0.5139 (2)
0.2083 (7)	0.4444 (5)	0.6667 (2)	0.3333 (6)	0.6667 (2)	0.625 (4)	0.6667 (2)
0.3611 (5)	0.3333 (6.5)	0.3333 (6.5)	0.4028 (3)	0.4028 (3)	0.5139 (1)	0.4028 (3)
0.3611 (7)	0.4444 (6)	1 (1)	0.5417 (5)	0.8056 (3.5)	0.9167 (2)	0.8056 (3.5)
0.2083 (7)	0.4444 (5)	0.4306 (6)	0.5 (4)	0.5139 (2.5)	0.5417 (1)	0.5139 (2.5)
0.25 (7)	0.4444 (5)	0.5139 (3)	0.4722 (4)	0.2917 (6)	0.5556 (1.5)	0.5556 (1.5)
0.1667 (7)	0.5556 (1)	0.4583 (6)	0.4722 (4)	0.4722 (4)	0.5417 (2)	0.4722 (4)
0.3194 (7)	0.3333 (6)	0.875 (1.5)	0.5833 (4.5)	0.625 (3)	0.875 (1.5)	0.5833 (4.5)
0.3194 (6.5)	0.3333 (5)	0.4028 (2.5)	0.4028 (2.5)	0.4028 (2.5)	0.3194 (6.5)	0.4028 (2.5)
0.25 (7)	0.3333 (6)	0.4722 (3.5)	0.4722 (3.5)	0.5556 (1.5)	0.3889 (5)	0.5556 (1.5)
0.3194 (7)	0.4444 (4)	0.4722 (3)	0.4306 (5)	0.3611 (6)	0.8333 (1)	0.5139 (2)
0.3194 (7)	0.3333 (6)	0.4306 (4)	0.4722 (2)	0.3611 (5)	0.5139 (1)	0.4444 (3)

Continuation of Table A.2

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.3194 (7)	0.3333 (6)	0.875 (1)	0.5833 (2.5)	0.5139 (4)	0.5833 (2.5)	0.4722 (5)
0.3611 (6)	0.3333 (7)	0.6944 (3)	0.4306 (5)	0.6944 (3)	0.7222 (1)	0.6944 (3)
0.3194 (7)	0.3333 (5.5)	0.3611 (4)	0.3333 (5.5)	0.5139 (1.5)	0.4722 (3)	0.5139 (1.5)
0.2083 (7)	0.4028 (3.5)	0.4028 (3.5)	0.5139 (1)	0.3194 (5.5)	0.3194 (5.5)	0.4306 (2)
0.4722 (1.5)	0.3333 (7)	0.4722 (1.5)	0.4306 (3.5)	0.3611 (5.5)	0.4306 (3.5)	0.3611 (5.5)
0.2083 (7)	0.3333 (6)	0.4306 (4)	0.4722 (2.5)	0.4722 (2.5)	0.5139 (1)	0.3611 (5)
0.3194 (7)	0.3333 (6)	0.625 (1)	0.5417 (4)	0.5833 (2.5)	0.5 (5)	0.5833 (2.5)
0.8472 (2)	0.3333 (7)	0.6944 (3)	0.5139 (5)	0.4722 (6)	0.9167 (1)	0.5556 (4)
0.3611 (4)	0.4444 (1)	0.4028 (2)	0.3194 (6)	0.3611 (4)	0.2778 (7)	0.3611 (4)
0.6944 (4)	0.3333 (7)	0.4444 (6)	0.5556 (5)	0.8056 (2.5)	0.8056 (2.5)	0.8472 (1)
0.2083 (7)	0.3333 (5.5)	0.6667 (1)	0.3333 (5.5)	0.625 (3)	0.625 (3)	0.625 (3)
0.3194 (7)	0.3333 (6)	0.4444 (4)	0.5556 (1.5)	0.4444 (4)	0.5556 (1.5)	0.4444 (4)
0.3611 (7)	0.4444 (6)	0.8889 (1)	0.4722 (5)	0.8472 (2.5)	0.8056 (4)	0.8472 (2.5)
0.1667 (7)	0.3333 (6)	0.6667 (2.5)	0.6667 (2.5)	0.5556 (4.5)	1 (1)	0.5556 (4.5)
0.6944 (4.5)	0.3333 (7)	0.7778 (3)	0.5139 (6)	0.8889 (1.5)	0.6944 (4.5)	0.8889 (1.5)
0.3611 (6)	0.3333 (7)	0.6667 (1)	0.5556 (3)	0.5833 (2)	0.5139 (4.5)	0.5139 (4.5)
0.4722 (1.5)	0.3333 (7)	0.4028 (4.5)	0.3611 (6)	0.4722 (1.5)	0.4306 (3)	0.4028 (4.5)
0.3611 (5)	0.3333 (6)	0.2917 (7)	0.4028 (4)	0.4722 (1.5)	0.4306 (3)	0.4722 (1.5)
0.2083 (7)	0.4444 (5)	0.7639 (4)	0.3889 (6)	0.8472 (1.5)	0.8056 (3)	0.8472 (1.5)
0.2083 (7)	0.4444 (6)	0.9583 (1)	0.5 (5)	0.5556 (2.5)	0.5139 (4)	0.5556 (2.5)

Continuation of Table A.2

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.3194 (7)	0.4444 (6)	0.5556 (1.5)	0.5556 (1.5)	0.5139 (3.5)	0.4722 (5)	0.5139 (3.5)
0.3194 (6)	0.2917 (7)	0.3611 (5)	0.5139 (1)	0.4028 (3.5)	0.4722 (2)	0.4028 (3.5)
0.1667 (7)	0.3333 (6)	0.3611 (3.5)	0.3611 (3.5)	0.3611 (3.5)	0.4722 (1)	0.3611 (3.5)
0.2083 (7)	0.3333 (6)	0.5139 (2)	0.4028 (5)	0.4444 (3.5)	0.5556 (1)	0.4444 (3.5)
0.6944 (4)	0.3333 (6.5)	0.7361 (3)	0.3333 (6.5)	0.5556 (5)	0.7639 (2)	0.8889 (1)
0.6944 (3)	0.4028 (6)	0.25 (7)	0.4722 (5)	0.8472 (1.5)	0.8472 (1.5)	0.5139 (4)
0.2083 (7)	0.4444 (6)	0.5139 (4)	0.5139 (4)	0.5556 (1.5)	0.5139 (4)	0.5556 (1.5)
0.3194 (7)	0.4444 (4.5)	0.5556 (2)	0.4444 (4.5)	0.4444 (4.5)	0.625 (1)	0.4444 (4.5)
0.2083 (7)	0.2917 (6)	0.4028 (3)	0.3611 (5)	0.4028 (3)	0.5139 (1)	0.4028 (3)
0.3194 (6.5)	0.4444 (4)	0.3194 (6.5)	0.3889 (5)	0.5139 (2)	0.5139 (2)	0.5139 (2)
0.3611 (7)	0.4444 (6)	0.4722 (4)	0.5139 (2)	0.4722 (4)	0.625 (1)	0.4722 (4)
0.1667 (7)	0.3333 (5.5)	0.3333 (5.5)	0.4306 (4)	0.5556 (1.5)	0.5139 (3)	0.5556 (1.5)
0.4028 (6)	0.3333 (7)	0.4722 (2)	0.4722 (2)	0.4306 (4.5)	0.4306 (4.5)	0.4722 (2)
0.3611 (7)	0.4444 (6)	0.8056 (1.5)	0.5556 (4)	0.5556 (4)	0.8056 (1.5)	0.5556 (4)
0.25 (6)	0.3333 (4)	0.2778 (5)	0.3889 (3)	0.4722 (2)	0.2361 (7)	0.5833 (1)
0.3194 (7)	0.3333 (6)	0.5556 (3)	0.5556 (3)	0.5556 (3)	0.5556 (3)	0.5556 (3)
0.3194 (7)	0.4444 (5.5)	0.4444 (5.5)	0.5139 (1.5)	0.5139 (1.5)	0.4722 (3.5)	0.4722 (3.5)
0.3194 (7)	0.4444 (4.5)	0.4444 (4.5)	0.4722 (3)	0.5556 (1.5)	0.4028 (6)	0.5556 (1.5)
0.3611 (5.5)	0.2917 (7)	0.4028 (2.5)	0.4028 (2.5)	0.4028 (2.5)	0.3611 (5.5)	0.4028 (2.5)
0.1667 (7)	0.4444 (5)	0.5139 (2)	0.5556 (1)	0.4444 (5)	0.4722 (3)	0.4444 (5)

Continuation of Table A.2

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.1667 (7)	0.4444 (5)	0.3611 (6)	0.5556 (3)	0.8056 (1.5)	0.8056 (1.5)	0.4722 (4)
0.3194 (7)	0.3333 (6)	0.5139 (1)	0.4444 (4)	0.4722 (2.5)	0.3889 (5)	0.4722 (2.5)
0.5417 (4)	0.4444 (6)	0.7778 (1.5)	0.4444 (6)	0.4444 (6)	0.6528 (3)	0.7778 (1.5)
0.3194 (7)	0.3333 (6)	0.8056 (2)	0.5556 (5)	0.8889 (1)	0.7639 (4)	0.7778 (3)
0.3194 (7)	0.4028 (2.5)	0.4028 (2.5)	0.3611 (5.5)	0.4028 (2.5)	0.3611 (5.5)	0.4028 (2.5)
0.3194 (7)	0.4444 (6)	0.4722 (5)	0.5556 (2)	0.5556 (2)	0.5139 (4)	0.5556 (2)
0.3194 (7)	0.3333 (6)	0.4444 (4.5)	0.5556 (1)	0.4444 (4.5)	0.4722 (3)	0.5139 (2)
0.2083 (7)	0.4444 (6)	0.8889 (2)	0.5417 (5)	0.6667 (3)	0.625 (4)	0.9583 (1)
0.3194 (7)	0.3333 (6)	0.5139 (3.5)	0.5139 (3.5)	0.5556 (1)	0.5139 (3.5)	0.5139 (3.5)
0.3194 (7)	0.3333 (6)	0.7222 (1)	0.625 (3)	0.625 (3)	0.5417 (5)	0.625 (3)
0.6944 (1)	0.3333 (6)	0.2917 (7)	0.3611 (5)	0.4028 (3)	0.4028 (3)	0.4028 (3)
0.3194 (7)	0.3333 (6)	0.5139 (4)	0.5139 (4)	0.5556 (1.5)	0.5139 (4)	0.5556 (1.5)
0.4722 (4.5)	0.3333 (7)	0.4722 (4.5)	0.5139 (2)	0.5139 (2)	0.4306 (6)	0.5139 (2)
0.3194 (7)	0.4028 (5.5)	0.4028 (5.5)	0.5139 (3)	0.625 (1.5)	0.4722 (4)	0.625 (1.5)
0.4722 (1.5)	0.2917 (7)	0.3889 (3.5)	0.3194 (6)	0.3611 (5)	0.3889 (3.5)	0.4722 (1.5)
0.2083 (7)	0.4444 (3)	0.3889 (6)	0.4306 (4.5)	0.4306 (4.5)	0.5 (2)	0.5139 (1)
0.6944 (1)	0.3333 (7)	0.5139 (3)	0.5833 (2)	0.4028 (5.5)	0.5 (4)	0.4028 (5.5)
0.3194 (7)	0.3333 (6)	0.5833 (4.5)	0.625 (3)	0.6667 (1.5)	0.5833 (4.5)	0.6667 (1.5)
0.3194 (6)	0.2917 (7)	0.5417 (1.5)	0.4722 (3)	0.4306 (4.5)	0.5417 (1.5)	0.4306 (4.5)
0.1667 (7)	0.4444 (2.5)	0.4444 (2.5)	0.4028 (5)	0.4444 (2.5)	0.3611 (6)	0.4444 (2.5)

Continuation of Table A.2

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.3611 (5.5)	0.4444 (1)	0.3611 (5.5)	0.4028 (3.5)	0.4028 (3.5)	0.4306 (2)	0.3194 (7)
0.6944 (2)	0.3333 (5.5)	0.6944 (2)	0.3333 (5.5)	0.4028 (4)	0.3194 (7)	0.6944 (2)
0.3611 (6)	0.3333 (7)	0.8472 (1)	0.5139 (3)	0.5139 (3)	0.4722 (5)	0.5139 (3)
0.3611 (7)	0.4028 (5)	0.4722 (2)	0.4722 (2)	0.4028 (5)	0.4722 (2)	0.4028 (5)
0.1667 (7)	0.4444 (4)	0.2361 (6)	0.3056 (5)	0.5417 (1.5)	0.5 (3)	0.5417 (1.5)
0.3611 (6)	0.4444 (5)	0.3333 (7)	0.5556 (2.5)	0.5556 (2.5)	0.5556 (2.5)	0.5556 (2.5)
0.3611 (5)	0.3333 (6.5)	0.5139 (4)	0.3333 (6.5)	0.5417 (3)	0.875 (1)	0.5833 (2)
0.4722 (2.5)	0.2917 (5.5)	0.25 (7)	0.2917 (5.5)	0.4722 (2.5)	0.4306 (4)	0.5139 (1)
0.8472 (1)	0.3333 (7)	0.4722 (5.5)	0.5139 (3)	0.4722 (5.5)	0.5139 (3)	0.5139 (3)
0.3194 (7)	0.3333 (5.5)	0.3333 (5.5)	0.6667 (1.5)	0.625 (3.5)	0.625 (3.5)	0.6667 (1.5)
0.3611 (3)	0.3333 (4)	0.2639 (6)	0.2361 (7)	0.4028 (1.5)	0.3194 (5)	0.4028 (1.5)
0.3194 (7)	0.3333 (6)	0.4028 (5)	0.5556 (1)	0.4444 (3)	0.4444 (3)	0.4444 (3)
0.2083 (7)	0.2917 (5.5)	0.5139 (1)	0.2917 (5.5)	0.3611 (3.5)	0.4722 (2)	0.3611 (3.5)
0.2083 (7)	0.4444 (3)	0.3333 (4)	0.3194 (5)	0.7361 (1.5)	0.2778 (6)	0.7361 (1.5)
0.3611 (6)	0.3333 (7)	0.4444 (3.5)	0.4444 (3.5)	0.5139 (1.5)	0.4306 (5)	0.5139 (1.5)
0.5139 (3)	0.3333 (7)	0.3889 (6)	0.4306 (5)	0.5139 (3)	0.8056 (1)	0.5139 (3)
0.1667 (7)	0.3333 (6)	0.5556 (2)	0.5139 (4)	0.5139 (4)	0.5833 (1)	0.5139 (4)
0.5139 (5)	0.3333 (7)	0.875 (1)	0.625 (2)	0.5139 (5)	0.5417 (3)	0.5139 (5)
0.3194 (7)	0.3333 (6)	0.4028 (2.5)	0.3611 (4.5)	0.5139 (1)	0.4028 (2.5)	0.3611 (4.5)
0.3194 (7)	0.3333 (6)	0.4444 (2)	0.3611 (5)	0.4444 (2)	0.4028 (4)	0.4444 (2)

Continuation of Table A.2

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.3611 (5)	0.3333 (6.5)	1 (1)	0.3333 (6.5)	0.5556 (4)	0.625 (3)	0.6667 (2)
0.3194 (7)	0.4028 (5.5)	0.5556 (1)	0.4306 (4)	0.4722 (3)	0.4028 (5.5)	0.5139 (2)
0.4722 (1)	0.3333 (3)	0.3611 (2)	0.3194 (5)	0.3194 (5)	0.2361 (7)	0.3194 (5)
0.4722 (1)	0.4444 (3.5)	0.4444 (3.5)	0.4444 (3.5)	0.4444 (3.5)	0.3611 (7)	0.4028 (6)
0.2083 (7)	0.4444 (5)	0.3611 (6)	0.4722 (3.5)	0.5833 (1)	0.5417 (2)	0.4722 (3.5)
0.3194 (7)	0.3333 (6)	0.7639 (1.5)	0.4306 (5)	0.5139 (4)	0.6944 (3)	0.7639 (1.5)
0.3194 (6)	0.4444 (3)	0.2917 (7)	0.4722 (1.5)	0.4722 (1.5)	0.3889 (5)	0.4306 (4)
0.4722 (3.5)	0.4444 (6)	0.3611 (7)	0.4722 (3.5)	0.4722 (3.5)	0.5139 (1)	0.4722 (3.5)
0.3194 (7)	0.4444 (4.5)	0.5139 (2.5)	0.5556 (1)	0.4028 (6)	0.5139 (2.5)	0.4444 (4.5)
0.3194 (6)	0.3333 (5)	0.2778 (7)	0.4722 (2.5)	0.5139 (1)	0.4306 (4)	0.4722 (2.5)
0.6944 (4)	0.3333 (7)	0.4722 (6)	0.5417 (5)	0.875 (1.5)	0.7222 (3)	0.875 (1.5)
0.3611 (5.5)	0.3333 (7)	0.3611 (5.5)	0.4722 (4)	0.5833 (1.5)	0.5417 (3)	0.5833 (1.5)
0.1667 (7)	0.5556 (1)	0.5 (3.5)	0.5 (3.5)	0.4306 (5.5)	0.5139 (2)	0.4306 (5.5)
0.3611 (6)	0.3333 (7)	0.4722 (1.5)	0.4722 (1.5)	0.4306 (4)	0.4306 (4)	0.4306 (4)
0.3194 (6)	0.3333 (3)	0.2917 (7)	0.3333 (3)	0.3333 (3)	0.3333 (3)	0.3333 (3)
0.1667 (7)	0.2917 (5)	0.2778 (6)	0.3611 (2)	0.3611 (2)	0.3194 (4)	0.3611 (2)
0.2083 (7)	0.4444 (4)	0.3611 (6)	0.4028 (5)	0.4722 (2)	0.4722 (2)	0.4722 (2)
0.1667 (7)	0.4444 (4)	0.3056 (6)	0.3472 (5)	0.5833 (2)	0.5833 (2)	0.5833 (2)
0.3194 (7)	0.3333 (6)	0.4028 (5)	0.4444 (3.5)	0.5556 (1)	0.5139 (2)	0.4444 (3.5)
0.3194 (6)	0.2917 (7)	0.4444 (4)	0.4722 (3)	0.5139 (1.5)	0.5139 (1.5)	0.3611 (5)

Continuation of Table A.2

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.2083 (7)	0.3333 (5)	0.2778 (6)	0.4028 (2.5)	0.4028 (2.5)	0.7639 (1)	0.3611 (4)
0.3611 (6)	0.3333 (7)	0.4444 (4)	0.4444 (4)	0.4444 (4)	0.5139 (2)	0.5556 (1)
0.3194 (7)	0.4444 (6)	0.4722 (5)	0.625 (3)	0.6667 (1.5)	0.5417 (4)	0.6667 (1.5)
0.3194 (7)	0.4444 (6)	0.9167 (2.5)	0.4583 (5)	0.8056 (4)	0.9167 (2.5)	1 (1)
0.1667 (7)	0.4028 (4)	0.4306 (2)	0.3472 (6)	0.4306 (2)	0.4306 (2)	0.3889 (5)
0.2083 (7)	0.4444 (5)	0.4306 (6)	0.5417 (2)	0.4722 (3.5)	0.875 (1)	0.4722 (3.5)
0.6944 (1)	0.3333 (4)	0.2917 (6.5)	0.3333 (4)	0.3333 (4)	0.625 (2)	0.2917 (6.5)
0.3611 (7)	0.4444 (2.5)	0.4028 (5)	0.4028 (5)	0.4028 (5)	0.6528 (1)	0.4444 (2.5)
0.3194 (7)	0.3333 (6)	0.5139 (3.5)	0.5556 (1.5)	0.5139 (3.5)	0.4306 (5)	0.5556 (1.5)
0.1667 (7)	0.3333 (6)	0.3611 (3.5)	0.3611 (3.5)	0.3611 (3.5)	0.3611 (3.5)	0.4028 (1)
0.3611 (6)	0.3333 (7)	0.5556 (2.5)	0.5556 (2.5)	0.5556 (2.5)	0.5417 (5)	0.5556 (2.5)
0.2083 (6)	0.2917 (2)	0.3333 (1)	0.2083 (6)	0.25 (3.5)	0.2083 (6)	0.25 (3.5)
0.3194 (7)	0.3333 (5.5)	0.7778 (1)	0.3333 (5.5)	0.6667 (2.5)	0.625 (4)	0.6667 (2.5)
0.6944 (5)	0.4444 (7)	0.8472 (2)	0.5139 (6)	0.8472 (2)	0.8056 (4)	0.8472 (2)
0.3194 (7)	0.3333 (6)	0.625 (1)	0.4028 (4.5)	0.4444 (3)	0.4028 (4.5)	0.5139 (2)
0.4722 (2)	0.3333 (5)	0.3611 (3.5)	0.3611 (3.5)	0.3194 (6.5)	0.5833 (1)	0.3194 (6.5)
0.4722 (3)	0.3333 (5)	0.3194 (6)	0.4028 (4)	0.2917 (7)	0.5417 (1)	0.5139 (2)
0.1667 (7)	0.4444 (6)	0.5139 (4.5)	0.5139 (4.5)	0.625 (1)	0.5833 (2.5)	0.5833 (2.5)
0.2083 (7)	0.3333 (6)	0.4306 (5)	0.4722 (4)	0.5833 (2)	0.5833 (2)	0.5833 (2)
0.6944 (1)	0.2917 (3.5)	0.2917 (3.5)	0.25 (6)	0.25 (6)	0.3611 (2)	0.25 (6)

Continuation of Table A.2

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.3611 (6)	0.3333 (7)	0.4028 (5)	0.5556 (3)	0.625 (1.5)	0.5417 (4)	0.625 (1.5)
0.6944 (1)	0.4444 (5)	0.4306 (6)	0.4167 (7)	0.5833 (2)	0.4583 (4)	0.5417 (3)
0.3194 (6)	0.3333 (3.5)	0.3333 (3.5)	0.3333 (3.5)	0.3333 (3.5)	0.4444 (1)	0.2917 (7)
0.3194 (6.5)	0.3333 (5)	0.3194 (6.5)	0.4306 (4)	0.4722 (2.5)	0.5417 (1)	0.4722 (2.5)
0.3194 (7)	0.3333 (5.5)	0.3333 (5.5)	0.5833 (2)	0.4722 (3)	0.3611 (4)	0.625 (1)
0.3194 (7)	0.4444 (6)	0.5556 (2.5)	0.5556 (2.5)	0.5556 (2.5)	0.5139 (5)	0.5556 (2.5)
0.6944 (1.5)	0.3333 (6.5)	0.625 (3)	0.3333 (6.5)	0.3611 (4.5)	0.6944 (1.5)	0.3611 (4.5)
0.1667 (7)	0.3333 (5)	0.2917 (6)	0.4028 (4)	0.4444 (2.5)	0.5139 (1)	0.4444 (2.5)
0.4722 (4)	0.3333 (7)	0.5139 (2.5)	0.5139 (2.5)	0.4028 (5.5)	0.625 (1)	0.4028 (5.5)
0.3611 (7)	0.4444 (6)	0.4722 (4.5)	0.5139 (3)	0.5556 (1.5)	0.4722 (4.5)	0.5556 (1.5)
0.2083 (7)	0.3333 (5.5)	0.4306 (4)	0.3333 (5.5)	0.6667 (1.5)	0.4722 (3)	0.6667 (1.5)
0.3194 (7)	0.4444 (6)	0.5417 (5)	0.5556 (3.5)	0.7778 (2)	0.9167 (1)	0.5556 (3.5)
0.3611 (4)	0.3333 (5.5)	0.2917 (7)	0.3333 (5.5)	0.4444 (2.5)	0.625 (1)	0.4444 (2.5)
0.1667 (7)	0.3333 (6)	0.5139 (5)	0.6667 (1)	0.625 (2.5)	0.5833 (4)	0.625 (2.5)
0.3611 (7)	0.4444 (4)	0.7361 (2)	0.4306 (5)	0.4722 (3)	0.3889 (6)	0.8056 (1)
0.3611 (6)	0.4444 (4)	0.3333 (7)	0.5556 (1.5)	0.5556 (1.5)	0.4028 (5)	0.5139 (3)
0.3194 (6)	0.2917 (7)	0.3611 (4.5)	0.4722 (2)	0.4722 (2)	0.3611 (4.5)	0.4722 (2)
0.3194 (6)	0.3333 (5)	0.4444 (3)	0.4722 (2)	0.2917 (7)	0.4306 (4)	0.5139 (1)
0.3611 (1)	0.3333 (2.5)	0.3333 (2.5)	0.2917 (5)	0.2917 (5)	0.25 (7)	0.2917 (5)
0.1667 (7)	0.3333 (5.5)	0.5833 (3)	0.3333 (5.5)	0.625 (1.5)	0.5 (4)	0.625 (1.5)

Continuation of Table A.2						
Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.3611 (2)	0.2917 (3.5)	0.1667 (6.5)	0.2917 (3.5)	0.2083 (5)	0.5 (1)	0.1667 (6.5)
0.3194 (7)	0.3333 (6)	0.3611 (4.5)	0.5556 (1)	0.3611 (4.5)	0.4722 (2.5)	0.4722 (2.5)
0.2083 (7)	0.3333 (2)	0.2917 (5)	0.3611 (1)	0.3194 (3.5)	0.2778 (6)	0.3194 (3.5)
0.5139 (4.5)	0.4444 (6.5)	0.4444 (6.5)	0.5556 (2)	0.5556 (2)	0.5139 (4.5)	0.5556 (2)
0.4722 (1)	0.2917 (7)	0.4028 (3)	0.3194 (6)	0.4028 (3)	0.3611 (5)	0.4028 (3)
0.4722 (2)	0.3333 (7)	0.4306 (4)	0.3889 (6)	0.4306 (4)	0.5 (1)	0.4306 (4)
0.4722 (1)	0.4444 (2.5)	0.4444 (2.5)	0.3611 (6.5)	0.4028 (4.5)	0.3611 (6.5)	0.4028 (4.5)
0.3194 (7)	0.4444 (6)	0.4722 (4.5)	0.4722 (4.5)	0.5139 (2.5)	0.8056 (1)	0.5139 (2.5)
0.5417 (4)	0.4444 (7)	0.5139 (5)	0.4722 (6)	0.5833 (2.5)	0.7639 (1)	0.5833 (2.5)
0.3611 (5)	0.4444 (3)	0.6528 (1)	0.4028 (4)	0.3194 (6)	0.5694 (2)	0.2778 (7)
0.3194 (7)	0.3333 (6)	0.5833 (3.5)	0.5556 (5)	0.625 (1.5)	0.5833 (3.5)	0.625 (1.5)
5.722	5.368	3.955	3.785	3.09	3.195	2.885

Table A.3: Repetition number prediction results for 200 resamples with ranks.

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.2083 (7)	0.3889 (5)	0.4444 (2)	0.4861 (1)	0.375 (6)	0.4028 (4)	0.4306 (3)
0.3611 (6)	0.3472 (7)	0.4861 (1)	0.4028 (3)	0.375 (4.5)	0.4444 (2)	0.375 (4.5)
0.3194 (5.5)	0.3194 (5.5)	0.3611 (3.5)	0.3611 (3.5)	0.4028 (2)	0.2917 (7)	0.4167 (1)
0.3611 (7)	0.4167 (2.5)	0.4028 (5)	0.4306 (1)	0.4028 (5)	0.4167 (2.5)	0.4028 (5)
0.4028 (6)	0.4167 (3.5)	0.4028 (6)	0.4306 (1.5)	0.4028 (6)	0.4306 (1.5)	0.4167 (3.5)
0.2639 (7)	0.3889 (6)	0.4722 (4)	0.4861 (3)	0.5278 (1.5)	0.4167 (5)	0.5278 (1.5)

Continuation of Table A.3

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.25 (7)	0.4028 (4)	0.4167 (2)	0.4167 (2)	0.375 (5.5)	0.4167 (2)	0.375 (5.5)
0.4306 (6)	0.375 (7)	0.4583 (4)	0.4722 (3)	0.4444 (5)	0.4861 (1.5)	0.4861 (1.5)
0.3611 (7)	0.375 (6)	0.4028 (5)	0.4722 (1)	0.4583 (2)	0.4306 (4)	0.4444 (3)
0.4167 (3.5)	0.3611 (6)	0.4306 (2)	0.4444 (1)	0.3333 (7)	0.4167 (3.5)	0.3889 (5)
0.3056 (7)	0.4722 (3)	0.4722 (3)	0.5 (1)	0.4167 (6)	0.4722 (3)	0.4444 (5)
0.2917 (7)	0.4306 (4.5)	0.4306 (4.5)	0.4583 (3)	0.4167 (6)	0.4722 (2)	0.4861 (1)
0.4167 (4)	0.4028 (5.5)	0.375 (7)	0.5 (1)	0.4583 (3)	0.4861 (2)	0.4028 (5.5)
0.4167 (3.5)	0.4167 (3.5)	0.3889 (6.5)	0.4306 (2)	0.4028 (5)	0.4583 (1)	0.3889 (6.5)
0.3472 (7)	0.4306 (3)	0.375 (5)	0.4167 (4)	0.4444 (1.5)	0.3611 (6)	0.4444 (1.5)
0.3889 (5.5)	0.4306 (4)	0.3889 (5.5)	0.4444 (2)	0.4444 (2)	0.4444 (2)	0.3333 (7)
0.4306 (2.5)	0.375 (6)	0.3333 (7)	0.4306 (2.5)	0.4167 (4)	0.4028 (5)	0.4722 (1)
0.3472 (6)	0.3194 (7)	0.4167 (3)	0.4306 (2)	0.4028 (4.5)	0.4444 (1)	0.4028 (4.5)
0.25 (7)	0.3472 (6)	0.4028 (2)	0.3889 (3.5)	0.4444 (1)	0.3889 (3.5)	0.375 (5)
0.4722 (2)	0.5 (1)	0.4444 (3)	0.4306 (4)	0.3472 (7)	0.4167 (5)	0.375 (6)
0.3611 (6)	0.375 (5)	0.4306 (2.5)	0.4444 (1)	0.3194 (7)	0.4306 (2.5)	0.4167 (4)
0.3333 (7)	0.4583 (2)	0.4306 (4)	0.4028 (5)	0.3889 (6)	0.4722 (1)	0.4444 (3)
0.4306 (5)	0.5 (1)	0.4167 (7)	0.4722 (2)	0.4306 (5)	0.4583 (3)	0.4306 (5)
0.4028 (7)	0.4167 (6)	0.4722 (2.5)	0.4722 (2.5)	0.4306 (4.5)	0.5139 (1)	0.4306 (4.5)
0.3056 (7)	0.3333 (6)	0.4583 (2)	0.4167 (5)	0.4444 (3.5)	0.4444 (3.5)	0.4861 (1)
0.3889 (4.5)	0.4444 (1)	0.3889 (4.5)	0.4028 (3)	0.3472 (7)	0.4306 (2)	0.3611 (6)

Continuation of Table A.3

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.4306 (6)	0.4444 (5)	0.4167 (7)	0.5 (2.5)	0.4861 (4)	0.5 (2.5)	0.5278 (1)
0.3056 (7)	0.375 (6)	0.4444 (3.5)	0.5139 (1)	0.4444 (3.5)	0.4167 (5)	0.4722 (2)
0.3889 (7)	0.4028 (6)	0.4583 (2)	0.4583 (2)	0.4306 (4.5)	0.4583 (2)	0.4306 (4.5)
0.3472 (7)	0.375 (6)	0.5139 (1)	0.4861 (2)	0.4583 (3.5)	0.4444 (5)	0.4583 (3.5)
0.3889 (4.5)	0.4583 (1.5)	0.3889 (4.5)	0.4306 (3)	0.3611 (6.5)	0.4583 (1.5)	0.3611 (6.5)
0.2222 (7)	0.4028 (5)	0.4722 (1)	0.4444 (2)	0.3889 (6)	0.4306 (3)	0.4167 (4)
0.2778 (7)	0.4028 (5)	0.4722 (1)	0.4306 (3)	0.3611 (6)	0.4583 (2)	0.4167 (4)
0.2361 (7)	0.3472 (6)	0.4028 (5)	0.4167 (3.5)	0.4306 (2)	0.4861 (1)	0.4167 (3.5)
0.3472 (7)	0.4722 (5)	0.5 (3)	0.5556 (1)	0.5 (3)	0.375 (6)	0.5 (3)
0.3472 (7)	0.4028 (6)	0.5 (1)	0.4722 (2.5)	0.4583 (4.5)	0.4583 (4.5)	0.4722 (2.5)
0.4583 (3)	0.4028 (5)	0.5139 (1)	0.4861 (2)	0.3611 (7)	0.4028 (5)	0.4028 (5)
0.2917 (6.5)	0.2917 (6.5)	0.4444 (2)	0.3889 (4)	0.4722 (1)	0.375 (5)	0.4028 (3)
0.3333 (7)	0.375 (6)	0.4167 (4.5)	0.4306 (3)	0.4167 (4.5)	0.4722 (1.5)	0.4722 (1.5)
0.2361 (7)	0.375 (5)	0.3889 (3.5)	0.4306 (2)	0.3611 (6)	0.4583 (1)	0.3889 (3.5)
0.4167 (2)	0.4028 (3.5)	0.4028 (3.5)	0.3889 (5.5)	0.375 (7)	0.4444 (1)	0.3889 (5.5)
0.4722 (1)	0.3889 (7)	0.4583 (2)	0.4444 (3.5)	0.4306 (5.5)	0.4444 (3.5)	0.4306 (5.5)
0.3056 (7)	0.4028 (2)	0.3889 (3)	0.4306 (1)	0.375 (4.5)	0.3472 (6)	0.375 (4.5)
0.3472 (6.5)	0.3472 (6.5)	0.4722 (2)	0.4722 (2)	0.4444 (4.5)	0.4722 (2)	0.4444 (4.5)
0.4722 (1)	0.3611 (6.5)	0.3611 (6.5)	0.4167 (3.5)	0.375 (5)	0.4583 (2)	0.4167 (3.5)
0.2917 (7)	0.3889 (4)	0.375 (6)	0.4028 (2)	0.4167 (1)	0.3889 (4)	0.3889 (4)

Continuation of Table A.3

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.3194 (7)	0.3889 (5)	0.3611 (6)	0.4444 (1)	0.4028 (3.5)	0.4028 (3.5)	0.4306 (2)
0.3056 (7)	0.4306 (5)	0.4167 (6)	0.4444 (4)	0.5 (1.5)	0.4722 (3)	0.5 (1.5)
0.3056 (7)	0.4028 (6)	0.4583 (5)	0.5278 (2.5)	0.5833 (1)	0.4861 (4)	0.5278 (2.5)
0.4722 (4.5)	0.4722 (4.5)	0.5278 (1)	0.4444 (6.5)	0.5 (2)	0.4444 (6.5)	0.4861 (3)
0.375 (7)	0.3889 (4.5)	0.4167 (1.5)	0.4167 (1.5)	0.3889 (4.5)	0.3889 (4.5)	0.3889 (4.5)
0.2917 (7)	0.4028 (4.5)	0.3472 (6)	0.4306 (3)	0.4861 (1)	0.4028 (4.5)	0.4444 (2)
0.375 (7)	0.4861 (1)	0.4722 (2)	0.4167 (5.5)	0.4444 (3)	0.4306 (4)	0.4167 (5.5)
0.3194 (7)	0.4306 (3)	0.4028 (5)	0.4861 (1)	0.4722 (2)	0.3889 (6)	0.4167 (4)
0.2778 (7)	0.3472 (6)	0.3611 (5)	0.375 (4)	0.4444 (1)	0.4306 (2)	0.4167 (3)
0.3611 (5)	0.4583 (1)	0.3611 (5)	0.3611 (5)	0.375 (3)	0.2917 (7)	0.4028 (2)
0.3611 (6.5)	0.4167 (4)	0.4028 (5)	0.3611 (6.5)	0.5 (1.5)	0.4861 (3)	0.5 (1.5)
0.3056 (7)	0.4306 (3)	0.4028 (5)	0.4444 (2)	0.4028 (5)	0.4583 (1)	0.4028 (5)
0.3194 (7)	0.4028 (5.5)	0.4861 (2.5)	0.5 (1)	0.4028 (5.5)	0.4861 (2.5)	0.4444 (4)
0.375 (7)	0.4306 (5)	0.4167 (6)	0.5 (1)	0.4444 (3.5)	0.4444 (3.5)	0.4583 (2)
0.3611 (7)	0.3889 (6)	0.4722 (4.5)	0.4722 (4.5)	0.5278 (1.5)	0.5139 (3)	0.5278 (1.5)
0.3472 (7)	0.375 (6)	0.4028 (4.5)	0.4583 (1)	0.4306 (2)	0.4028 (4.5)	0.4167 (3)
0.4583 (1.5)	0.375 (6)	0.4444 (3)	0.3611 (7)	0.4583 (1.5)	0.4028 (5)	0.4306 (4)
0.3194 (6.5)	0.4306 (2)	0.4167 (3)	0.4861 (1)	0.3333 (4.5)	0.3194 (6.5)	0.3333 (4.5)
0.2361 (7)	0.3889 (3)	0.4444 (1.5)	0.375 (4.5)	0.375 (4.5)	0.3472 (6)	0.4444 (1.5)
0.4167 (3.5)	0.3889 (5)	0.4861 (1)	0.3611 (7)	0.4167 (3.5)	0.375 (6)	0.4583 (2)

Continuation of Table A.3

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.3194 (7)	0.3889 (5.5)	0.4444 (2)	0.4722 (1)	0.4028 (4)	0.3889 (5.5)	0.4306 (3)
0.375 (6.5)	0.375 (6.5)	0.4583 (2.5)	0.4444 (4)	0.5139 (1)	0.4306 (5)	0.4583 (2.5)
0.2917 (7)	0.3889 (4.5)	0.3889 (4.5)	0.4306 (1.5)	0.3889 (4.5)	0.4306 (1.5)	0.3889 (4.5)
0.375 (6.5)	0.3889 (5)	0.4167 (4)	0.4861 (1)	0.4444 (2)	0.375 (6.5)	0.4306 (3)
0.3056 (7)	0.4306 (4)	0.4444 (3)	0.4583 (2)	0.4167 (5.5)	0.5139 (1)	0.4167 (5.5)
0.3194 (7)	0.4167 (3.5)	0.4444 (1)	0.4306 (2)	0.4167 (3.5)	0.375 (6)	0.3889 (5)
0.4306 (4)	0.5139 (2.5)	0.5278 (1)	0.5139 (2.5)	0.375 (6)	0.4167 (5)	0.3611 (7)
0.2361 (7)	0.3889 (4)	0.375 (5)	0.4722 (1.5)	0.3611 (6)	0.4722 (1.5)	0.4028 (3)
0.3611 (7)	0.4306 (4.5)	0.4167 (6)	0.4722 (3)	0.5 (1.5)	0.5 (1.5)	0.4306 (4.5)
0.3194 (7)	0.4167 (3.5)	0.4028 (5)	0.4167 (3.5)	0.4444 (2)	0.375 (6)	0.5 (1)
0.375 (4)	0.3472 (6.5)	0.4583 (1)	0.3889 (2)	0.375 (4)	0.3472 (6.5)	0.375 (4)
0.5139 (1)	0.3611 (7)	0.4167 (5)	0.4444 (2)	0.4028 (6)	0.4306 (3.5)	0.4306 (3.5)
0.2917 (7)	0.3889 (5)	0.4028 (3.5)	0.4444 (1)	0.3472 (6)	0.4028 (3.5)	0.4167 (2)
0.375 (5.5)	0.4028 (4)	0.4167 (2)	0.4167 (2)	0.4167 (2)	0.3333 (7)	0.375 (5.5)
0.4167 (4)	0.4583 (1)	0.4306 (3)	0.4444 (2)	0.2778 (7)	0.3056 (5.5)	0.3056 (5.5)
0.3056 (7)	0.3194 (6)	0.3472 (5)	0.4306 (2.5)	0.4444 (1)	0.3611 (4)	0.4306 (2.5)
0.375 (6.5)	0.4028 (4)	0.4444 (2)	0.4028 (4)	0.4028 (4)	0.375 (6.5)	0.4722 (1)
0.2778 (7)	0.5139 (1)	0.4583 (3)	0.4722 (2)	0.4306 (4.5)	0.4306 (4.5)	0.3889 (6)
0.3611 (7)	0.4306 (2)	0.4028 (4.5)	0.4444 (1)	0.3889 (6)	0.4028 (4.5)	0.4167 (3)
0.2917 (7)	0.4583 (4)	0.4722 (3)	0.5139 (1)	0.4861 (2)	0.4444 (5)	0.4167 (6)

Continuation of Table A.3

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.4028 (6)	0.4167 (4)	0.4444 (1)	0.4167 (4)	0.4306 (2)	0.3889 (7)	0.4167 (4)
0.4583 (1)	0.375 (6.5)	0.4167 (4.5)	0.4306 (2.5)	0.375 (6.5)	0.4167 (4.5)	0.4306 (2.5)
0.25 (7)	0.3472 (4)	0.4306 (2)	0.4444 (1)	0.3333 (5.5)	0.4167 (3)	0.3333 (5.5)
0.3056 (7)	0.4028 (1)	0.3611 (3.5)	0.3889 (2)	0.3611 (3.5)	0.3472 (5)	0.3333 (6)
0.3611 (7)	0.4028 (5)	0.375 (6)	0.4167 (4)	0.4444 (2.5)	0.4583 (1)	0.4444 (2.5)
0.2917 (7)	0.4167 (5)	0.3889 (6)	0.4306 (3.5)	0.4306 (3.5)	0.4722 (1)	0.4583 (2)
0.3194 (6.5)	0.3194 (6.5)	0.3472 (4.5)	0.3611 (2.5)	0.3472 (4.5)	0.4583 (1)	0.3611 (2.5)
0.2917 (7)	0.375 (5)	0.3889 (4)	0.3611 (6)	0.4028 (2.5)	0.5 (1)	0.4028 (2.5)
0.3194 (7)	0.4306 (3)	0.4306 (3)	0.5278 (1)	0.375 (6)	0.4306 (3)	0.3889 (5)
0.3333 (7)	0.3889 (6)	0.4167 (5)	0.4583 (2.5)	0.4444 (4)	0.5417 (1)	0.4583 (2.5)
0.3889 (4)	0.4306 (1)	0.375 (5.5)	0.3333 (7)	0.4028 (2.5)	0.375 (5.5)	0.4028 (2.5)
0.3333 (7)	0.4306 (2)	0.375 (4.5)	0.4444 (1)	0.3472 (6)	0.3889 (3)	0.375 (4.5)
0.3472 (6)	0.3333 (7)	0.375 (3)	0.4028 (1)	0.3889 (2)	0.3611 (4.5)	0.3611 (4.5)
0.3611 (6)	0.4028 (2)	0.375 (4)	0.4444 (1)	0.375 (4)	0.2917 (7)	0.375 (4)
0.3333 (7)	0.4444 (1)	0.3611 (6)	0.4306 (2)	0.375 (5)	0.4028 (3.5)	0.4028 (3.5)
0.2778 (7)	0.3611 (5.5)	0.3889 (3.5)	0.3611 (5.5)	0.3889 (3.5)	0.4028 (2)	0.4306 (1)
0.3472 (5)	0.3194 (6.5)	0.3194 (6.5)	0.3889 (3)	0.4028 (1.5)	0.3611 (4)	0.4028 (1.5)
0.3333 (6)	0.375 (3.5)	0.375 (3.5)	0.3889 (1.5)	0.3333 (6)	0.3889 (1.5)	0.3333 (6)
0.375 (5)	0.4028 (2.5)	0.3194 (7)	0.4028 (2.5)	0.4028 (2.5)	0.3333 (6)	0.4028 (2.5)
0.3194 (7)	0.4167 (2)	0.4028 (3)	0.4306 (1)	0.3611 (4.5)	0.3472 (6)	0.3611 (4.5)

Continuation of Table A.3

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.4444 (2)	0.3611 (7)	0.4028 (6)	0.4167 (4.5)	0.4167 (4.5)	0.4583 (1)	0.4306 (3)
0.3889 (6)	0.4028 (5)	0.4167 (3.5)	0.4306 (2)	0.4583 (1)	0.375 (7)	0.4167 (3.5)
0.2917 (7)	0.375 (6)	0.4444 (2)	0.4722 (1)	0.4167 (4)	0.4167 (4)	0.4167 (4)
0.375 (7)	0.4167 (4.5)	0.4167 (4.5)	0.3889 (6)	0.4444 (2.5)	0.4722 (1)	0.4444 (2.5)
0.2917 (7)	0.4028 (5)	0.4306 (4)	0.4861 (1)	0.4444 (3)	0.375 (6)	0.4722 (2)
0.3194 (7)	0.3611 (5.5)	0.3889 (4)	0.4444 (1)	0.4306 (2.5)	0.3611 (5.5)	0.4306 (2.5)
0.3333 (7)	0.4444 (4)	0.4167 (6)	0.4583 (2)	0.4583 (2)	0.4583 (2)	0.4306 (5)
0.3194 (7)	0.4444 (6)	0.4722 (4.5)	0.4722 (4.5)	0.5278 (1.5)	0.5139 (3)	0.5278 (1.5)
0.2778 (7)	0.3472 (5)	0.4444 (2)	0.4583 (1)	0.3472 (5)	0.4167 (3)	0.3472 (5)
0.2917 (7)	0.4167 (4.5)	0.4167 (4.5)	0.4444 (1.5)	0.4028 (6)	0.4306 (3)	0.4444 (1.5)
0.3056 (7)	0.4444 (2.5)	0.4444 (2.5)	0.5 (1)	0.4167 (4)	0.375 (6)	0.4028 (5)
0.2917 (7)	0.3889 (2)	0.4306 (1)	0.375 (3)	0.3194 (5.5)	0.3611 (4)	0.3194 (5.5)
0.2222 (7)	0.3472 (4.5)	0.375 (3)	0.4167 (1)	0.3333 (6)	0.4028 (2)	0.3472 (4.5)
0.3611 (7)	0.4167 (5.5)	0.5 (1)	0.4444 (4)	0.4167 (5.5)	0.4583 (2.5)	0.4583 (2.5)
0.3333 (7)	0.3889 (6)	0.5278 (1)	0.4167 (5)	0.4861 (2)	0.4583 (4)	0.4722 (3)
0.4167 (5)	0.5139 (1)	0.3889 (6.5)	0.4722 (3)	0.4861 (2)	0.3889 (6.5)	0.4444 (4)
0.4028 (5.5)	0.4028 (5.5)	0.4861 (3)	0.4444 (4)	0.3889 (7)	0.5694 (1)	0.5 (2)
0.2361 (7)	0.375 (6)	0.4583 (3)	0.4444 (5)	0.4583 (3)	0.4583 (3)	0.4722 (1)
0.2778 (7)	0.375 (5.5)	0.4167 (3)	0.4028 (4)	0.4444 (1)	0.4306 (2)	0.375 (5.5)
0.3611 (7)	0.3889 (6)	0.5 (3)	0.5139 (2)	0.4306 (4.5)	0.4306 (4.5)	0.5278 (1)

Continuation of Table A.3

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.3056 (7)	0.375 (5)	0.4167 (4)	0.5 (1)	0.4444 (2.5)	0.3611 (6)	0.4444 (2.5)
0.3472 (7)	0.3889 (2.5)	0.3889 (2.5)	0.3611 (6)	0.375 (4.5)	0.375 (4.5)	0.4444 (1)
0.3611 (5.5)	0.3611 (5.5)	0.3889 (3.5)	0.3889 (3.5)	0.4583 (1.5)	0.3056 (7)	0.4583 (1.5)
0.4306 (3.5)	0.3472 (7)	0.4583 (1.5)	0.4028 (5.5)	0.4028 (5.5)	0.4306 (3.5)	0.4583 (1.5)
0.4028 (6.5)	0.4306 (5)	0.4722 (3)	0.4028 (6.5)	0.4861 (1)	0.4722 (3)	0.4722 (3)
0.3889 (5)	0.4583 (3)	0.3611 (7)	0.3889 (5)	0.5 (1)	0.3889 (5)	0.4722 (2)
0.3194 (7)	0.3889 (5)	0.4167 (4)	0.4444 (3)	0.4861 (1.5)	0.375 (6)	0.4861 (1.5)
0.3056 (7)	0.4028 (6)	0.4444 (3.5)	0.4444 (3.5)	0.4722 (1.5)	0.4306 (5)	0.4722 (1.5)
0.2639 (7)	0.3472 (6)	0.4444 (1.5)	0.4444 (1.5)	0.4167 (3)	0.375 (5)	0.4028 (4)
0.3056 (7)	0.4444 (2)	0.4583 (1)	0.3472 (4)	0.3194 (5.5)	0.4028 (3)	0.3194 (5.5)
0.3611 (7)	0.4167 (3)	0.4028 (4)	0.4583 (1)	0.375 (6)	0.4444 (2)	0.3889 (5)
0.3333 (7)	0.3889 (4)	0.4444 (1.5)	0.4444 (1.5)	0.3472 (5.5)	0.4167 (3)	0.3472 (5.5)
0.3889 (7)	0.4167 (6)	0.4306 (4.5)	0.4444 (3)	0.4722 (1.5)	0.4306 (4.5)	0.4722 (1.5)
0.3472 (6.5)	0.3472 (6.5)	0.4722 (2)	0.4167 (5)	0.4722 (2)	0.4444 (4)	0.4722 (2)
0.3056 (7)	0.4306 (4.5)	0.4722 (2)	0.5 (1)	0.4306 (4.5)	0.4306 (4.5)	0.4306 (4.5)
0.375 (4.5)	0.375 (4.5)	0.375 (4.5)	0.3611 (7)	0.375 (4.5)	0.4444 (1)	0.4167 (2)
0.3194 (7)	0.4167 (4.5)	0.4028 (6)	0.4306 (3)	0.4722 (1)	0.4167 (4.5)	0.4583 (2)
0.3194 (7)	0.3472 (5.5)	0.4306 (1)	0.3472 (5.5)	0.3889 (2.5)	0.3889 (2.5)	0.3611 (4)
0.2917 (7)	0.4028 (4)	0.3889 (5)	0.4444 (3)	0.4583 (1.5)	0.375 (6)	0.4583 (1.5)
0.3472 (7)	0.4583 (2)	0.4444 (3)	0.4861 (1)	0.4028 (5.5)	0.4167 (4)	0.4028 (5.5)

Continuation of Table A.3

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.2917 (7)	0.3472 (6)	0.3611 (5)	0.3889 (4)	0.4028 (2.5)	0.4167 (1)	0.4028 (2.5)
0.3194 (6)	0.4028 (2)	0.3056 (7)	0.4306 (1)	0.3889 (3)	0.3472 (4)	0.3333 (5)
0.3333 (6.5)	0.3333 (6.5)	0.4028 (3)	0.4028 (3)	0.4028 (3)	0.4444 (1)	0.375 (5)
0.3889 (7)	0.4167 (5)	0.5139 (1)	0.4028 (6)	0.4444 (4)	0.4722 (3)	0.4861 (2)
0.2361 (7)	0.4167 (5.5)	0.4583 (2.5)	0.4167 (5.5)	0.4583 (2.5)	0.4583 (2.5)	0.4583 (2.5)
0.4028 (5.5)	0.4583 (1)	0.4167 (3)	0.4167 (3)	0.4167 (3)	0.4028 (5.5)	0.375 (7)
0.2778 (7)	0.3889 (5)	0.375 (6)	0.4306 (1)	0.4028 (3.5)	0.4167 (2)	0.4028 (3.5)
0.3194 (7)	0.4028 (4)	0.5 (1)	0.4583 (2)	0.4028 (4)	0.3889 (6)	0.4028 (4)
0.3056 (7)	0.3889 (6)	0.4583 (3)	0.4861 (1)	0.4583 (3)	0.4444 (5)	0.4583 (3)
0.3472 (7)	0.4167 (4.5)	0.4167 (4.5)	0.375 (6)	0.4583 (1.5)	0.4306 (3)	0.4583 (1.5)
0.3611 (7)	0.4444 (4)	0.4444 (4)	0.3889 (6)	0.5278 (1)	0.4444 (4)	0.4722 (2)
0.4167 (4.5)	0.4167 (4.5)	0.4167 (4.5)	0.4167 (4.5)	0.4722 (1)	0.4028 (7)	0.4444 (2)
0.375 (3.5)	0.4028 (1)	0.375 (3.5)	0.3889 (2)	0.3194 (7)	0.3333 (6)	0.3472 (5)
0.4861 (1)	0.375 (7)	0.4028 (5.5)	0.4028 (5.5)	0.4722 (2)	0.4444 (3.5)	0.4444 (3.5)
0.2361 (7)	0.375 (4.5)	0.375 (4.5)	0.4167 (1)	0.3889 (2.5)	0.3333 (6)	0.3889 (2.5)
0.3889 (5)	0.4028 (4)	0.4722 (1)	0.4444 (2)	0.375 (6)	0.3472 (7)	0.4167 (3)
0.3056 (7)	0.4028 (3.5)	0.4167 (2)	0.4306 (1)	0.3611 (5.5)	0.4028 (3.5)	0.3611 (5.5)
0.375 (7)	0.4306 (6)	0.4583 (2)	0.4583 (2)	0.4444 (4.5)	0.4583 (2)	0.4444 (4.5)
0.3611 (7)	0.375 (5.5)	0.4167 (4)	0.4722 (1)	0.375 (5.5)	0.4444 (2.5)	0.4444 (2.5)
0.3194 (7)	0.3611 (6)	0.4306 (1.5)	0.4306 (1.5)	0.4167 (3.5)	0.4167 (3.5)	0.3889 (5)

Continuation of Table A.3

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.4028 (6)	0.3889 (7)	0.4167 (5)	0.4861 (3)	0.5 (1.5)	0.4306 (4)	0.5 (1.5)
0.3194 (7)	0.3333 (6)	0.375 (4.5)	0.375 (4.5)	0.4444 (2)	0.5 (1)	0.4167 (3)
0.3056 (7)	0.3333 (6)	0.3611 (4)	0.375 (2)	0.375 (2)	0.3472 (5)	0.375 (2)
0.4444 (1)	0.375 (4.5)	0.4028 (3)	0.4167 (2)	0.375 (4.5)	0.3472 (7)	0.3611 (6)
0.3333 (7)	0.3889 (4.5)	0.3889 (4.5)	0.3472 (6)	0.4583 (1)	0.4028 (3)	0.4167 (2)
0.4167 (1.5)	0.4028 (3.5)	0.4028 (3.5)	0.4167 (1.5)	0.3194 (7)	0.3889 (5)	0.3472 (6)
0.2222 (7)	0.4444 (3)	0.4444 (3)	0.4583 (1)	0.4028 (5.5)	0.4444 (3)	0.4028 (5.5)
0.3889 (4)	0.5 (1)	0.3472 (6.5)	0.3472 (6.5)	0.4444 (2)	0.375 (5)	0.4306 (3)
0.2778 (7)	0.4167 (4)	0.3472 (6)	0.4722 (2)	0.375 (5)	0.5139 (1)	0.4306 (3)
0.4167 (6.5)	0.4444 (4.5)	0.4444 (4.5)	0.5 (1)	0.4583 (3)	0.4861 (2)	0.4167 (6.5)
0.2639 (7)	0.3611 (6)	0.375 (5)	0.4583 (2)	0.3889 (4)	0.4167 (3)	0.4722 (1)
0.4167 (5.5)	0.4444 (3)	0.4167 (5.5)	0.375 (7)	0.4861 (2)	0.4306 (4)	0.5139 (1)
0.4306 (3.5)	0.3472 (7)	0.4306 (3.5)	0.4444 (2)	0.375 (6)	0.4028 (5)	0.4583 (1)
0.4167 (6.5)	0.4583 (3)	0.4167 (6.5)	0.4583 (3)	0.4722 (1)	0.4583 (3)	0.4306 (5)
0.3194 (7)	0.4028 (6)	0.4444 (2)	0.4306 (4)	0.4444 (2)	0.4167 (5)	0.4444 (2)
0.3194 (7)	0.4167 (4)	0.3889 (5.5)	0.4583 (1)	0.3889 (5.5)	0.4306 (2.5)	0.4306 (2.5)
0.3611 (6)	0.3472 (7)	0.4583 (1)	0.4167 (3.5)	0.4167 (3.5)	0.4167 (3.5)	0.4167 (3.5)
0.3194 (7)	0.4167 (6)	0.4722 (1.5)	0.4444 (3)	0.4306 (4.5)	0.4722 (1.5)	0.4306 (4.5)
0.3889 (5.5)	0.4444 (1)	0.4167 (2)	0.375 (7)	0.4028 (3.5)	0.4028 (3.5)	0.3889 (5.5)
0.3056 (7)	0.3889 (5)	0.4028 (4)	0.4583 (3)	0.4722 (1.5)	0.3472 (6)	0.4722 (1.5)

Continuation of Table A.3

Random	SVM	SVM+l	SVM+r	MLP	MLP+t	MLP+r
0.3611 (7)	0.3889 (5)	0.4722 (3.5)	0.4861 (2)	0.375 (6)	0.4722 (3.5)	0.5 (1)
0.3333 (7)	0.375 (4)	0.3472 (6)	0.3611 (5)	0.4167 (1)	0.3889 (3)	0.4028 (2)
0.2361 (7)	0.375 (5)	0.4028 (4)	0.4167 (2)	0.4167 (2)	0.3472 (6)	0.4167 (2)
0.4167 (4.5)	0.4167 (4.5)	0.4444 (3)	0.4861 (1)	0.375 (6)	0.4583 (2)	0.3611 (7)
0.3889 (6.5)	0.3889 (6.5)	0.4028 (5)	0.4861 (1)	0.4444 (2)	0.4167 (3.5)	0.4167 (3.5)
0.2639 (7)	0.375 (3.5)	0.3472 (5.5)	0.375 (3.5)	0.4028 (1)	0.3472 (5.5)	0.3889 (2)
0.3889 (5)	0.4444 (3)	0.375 (6)	0.4861 (1)	0.4167 (4)	0.4583 (2)	0.3472 (7)
0.2778 (7)	0.375 (6)	0.5278 (1)	0.4583 (5)	0.4861 (2)	0.4722 (3.5)	0.4722 (3.5)
0.3056 (7)	0.4583 (6)	0.4861 (3.5)	0.4861 (3.5)	0.5 (1.5)	0.4722 (5)	0.5 (1.5)
0.3472 (6.5)	0.4167 (3.5)	0.4167 (3.5)	0.3889 (5)	0.3472 (6.5)	0.4722 (1)	0.4306 (2)
0.2917 (7)	0.4167 (6)	0.4722 (3)	0.5 (2)	0.4444 (4.5)	0.5278 (1)	0.4444 (4.5)
0.3611 (6.5)	0.3611 (6.5)	0.3889 (5)	0.4444 (1.5)	0.4028 (3.5)	0.4444 (1.5)	0.4028 (3.5)
0.2917 (7)	0.4583 (1)	0.4167 (3.5)	0.4028 (6)	0.4167 (3.5)	0.4167 (3.5)	0.4167 (3.5)
0.375 (6)	0.375 (6)	0.375 (6)	0.4583 (1)	0.4028 (3)	0.4028 (3)	0.4028 (3)
6.122	4.54	3.69	2.852	3.678	3.682	3.435

École polytechnique de Louvain