
# Machine-Learning to Promote Better Practices of Low-Cost Wearable Devices in Tremor Diseases

**Julián David Loaiza Duque**

Director: Dr. Antonio J. Sánchez Egea

Co-director: Dr. Hernán A. González Rojas

*A dissertation submitted to Universitat Politècnica de Catalunya for the degree of Doctor of Philosophy in Mechanical, Fluids and Aerospace Engineering*

Escola Politècnica Superior d'Enginyeria de Vilanova i la Geltrú

Universitat Politècnica de Catalunya - BarcelonaTech

Barcelona, September 2023

# Abstract

The most common tremor syndromes worldwide are Parkinson's disease and essential tremor. Medical evaluation by neurologists remains the gold standard for diagnosing both movement disorders. Specific protocols have been established for these assessments, such as the Unified Parkinson's Disease Rating Scale and the Fahn-Tolosa-Marin scale for Essential Tremor. However, the accuracy of this practice depends primarily on the experience and skill of the treating physician or specialist. An effective tool for diagnosing Parkinson's disease is $^{123}$I-FP-CIT SPECT[1] . Although this test is expensive, access to this technology is limited or non-existent in many countries, particularly in developing regions. In addition, the patient must be compatible with the radiopharmaceutical tracer used in the test, making it an invasive procedure.

In this context, there are no low-cost methods or techniques that ensure accurate, safe, and reproducible differentiation between these two diseases. Diagnostic methods have been developed that use inertial measurement units to record and analyze upper extremity tremors. However, implementing these methods in clinical practice is challenging due to the multiple maneuvers required, staff training, and the availability of specialized equipment to perform the distraction tests. This research focuses on developing a low-cost, non-invasive, user-friendly decision support tool that integrates wearable devices with built-in inertial sensors and machine learning-based classification models. The goal is to achieve a fast and straightforward real-time differential classification of these movement disorders.

An initial kinematic analysis of hand tremor recordings from various subjects was performed in the MATLAB development environment. These recordings were obtained from a database of previous research, including data from healthy subjects and patients diagnosed by movement disorder specialists. Discriminative features of hand tremors in the frequency spectrum were extracted from linear acceleration and angular velocity signals acquired by accelerometers and gyroscopes from

[1] $^{123}$I-FP-CIT SPECT (DaTScan) is a complementary tool in the differential diagnosis of patients with incomplete or uncertain Parkinsonism. It uses a radiopharmaceutical tracer that provides information about dopamine transporters.

smartphones and inertial measurement units. The next phase involved the development of machine learning models. Two models were created: one to discriminate between physiological and pathological tremors and another to discriminate between pathological tremors. Several performance parameters were considered, such as the sensor axes used, the signal segments processed, the frequency range analyzed, the kinematic features used, the train/test ratio, the classification method, and its hyperparameters. These variables were crucial to optimize the identification of tremor types. The results of this initial research phase have been published in three papers, demonstrating that combining linear acceleration and angular velocity information from hand tremor signals with machine learning has significant potential for tremor-type classification.

Subsequently, the decision support tool was conceptualized as a modular system consisting of three basic components:1) a mobile application for data acquisition and visualization of classification results; 2) a RESTful API that serves as a web server/backend that receives, processes, and classifies the data; and 3) a database for records of diagnosed subjects. The mobile application, developed in Kotlin for Android devices, allows the input of demographic data and the recording of signals for subsequent classification. The algorithms used for signal analysis and training new models were implemented in Python. The same programming language was used to create a RESTful API using the Flask framework to deploy the developed algorithms and models. Additionally, this server can store kinematic records of patients with confirmed diagnoses in a PostgreSQL database. Expanding this database will improve the machine learning models and enhance the capabilities of the tool. The development process and operation of the resulting application have been licensed and registered as *TremorSoft*.

# Resumen

Los síndromes de temblor más prevalentes en todo el mundo son la enfermedad de Parkinson y el Temblor Esencial. La evaluación médica por neurólogos sigue siendo el patrón de referencia para diagnosticar ambos trastornos del movimiento. Se han adoptado protocolos específicos para estas evaluaciones, como la Escala Unificada de Calificación de la Enfermedad de Parkinson y la escala Fahn-Tolosa-Marin para el Temblor Esencial. Sin embargo, la precisión de esta práctica depende principalmente de la experiencia y las habilidades del médico tratante o especialista. Una herramienta eficaz para diagnosticar la enfermedad de Parkinson es la $^{123}$I-FP-CIT SPECT[2] Aunque esta prueba es costosa, en muchos países, especialmente en los que se encuentran en regiones en vías de desarrollo, el acceso a esta tecnología es limitado o inexistente. Además, el paciente debe ser compatible con el trazador radiofarmacéutico utilizado en la prueba, lo que la convierte en un procedimiento invasivo.

En este contexto, no existen métodos o técnicas de bajo coste que garanticen una diferenciación precisa, segura y reproducible entre estos dos trastornos. Se han desarrollado métodos de diagnóstico que emplean unidades de medición inercial para registrar y analizar los temblores de las extremidades superiores. Sin embargo, la implementación de estos métodos en la práctica clínica presenta retos debido a las múltiples maniobras requeridas, la necesidad de formación del personal y la disponibilidad de equipos especializados para realizar las pruebas de distracción. Esta investigación se ha centrado en el desarrollo de una herramienta de apoyo al diagnóstico cuantitativo que sea rentable, no invasiva, fácil de usar e integre dispositivos wearables con sensores inerciales y modelos de clasificación basados en machine learning. El objetivo es lograr una clasificación diferencial rápida y sencilla en tiempo real de estos trastornos del movimiento.

En el entorno de desarrollo MATLAB se realizó un análisis cinemático inicial de los registros del temblor de la mano de varios sujetos. Estos registros procedían de una base de datos de investi-

[2] $^{123}$I-FP-CIT SPECT (DaTScan) es una herramienta complementaria en el diagnóstico diferencial de pacientes con parkinsonismo incompleto o incierto. Utiliza un trazador radiofarmacéutico que proporciona información sobre los transportadores de dopamina.

gaciones anteriores, que incluía datos de sujetos sanos y pacientes diagnosticados por especialistas en trastornos del movimiento. Se extrajeron características discriminatorias de los temblores de la mano en el espectro de frecuencias a partir de señales de aceleración lineal y velocidad angular adquiridas mediante acelerómetros y giroscopios de smartphones y unidades de medición inercial. La fase siguiente consistió en el desarrollo de modelos de aprendizaje automático. Se crearon dos modelos: uno para distinguir entre temblores fisiológicos y patológicos y otro para diferenciar entre temblores patológicos. Se tuvieron en cuenta diversos parámetros de rendimiento, como los ejes de los sensores utilizados, los segmentos de señal procesados, el rango de frecuencias analizado, las características cinemáticas utilizadas, las relaciones entrenamiento/prueba, el método de clasificación y sus hiperparámetros. Estas variables fueron fundamentales para optimizar la identificación de los tipos de temblor. Los resultados de esta fase inicial de la investigación se han publicado en tres artículos, demostrando que la combinación de la aceleración lineal y la información de la señal de velocidad angular de los temblores de la mano con el aprendizaje automático tiene un potencial significativo para la clasificación del tipo de temblor.

Posteriormente, la herramienta se conceptualizó como un sistema modular compuesto por tres componentes fundamentales: 1) Una aplicación móvil para la adquisición de datos y visualización de los resultados de la clasificación; 2) Una API RESTful, que sirve como servidor web/back-end que recibe, procesa y clasifica los datos; y 3) Una base de datos para los registros de los sujetos diagnosticados. Desarrollada en Kotlin para dispositivos Android, la aplicación móvil permite la introducción de datos demográficos y el registro de señales para su posterior clasificación. Los algoritmos utilizados para el análisis de señales y el entrenamiento de nuevos modelos se implementaron en Python. El mismo lenguaje de programación se empleó para crear una API RESTful utilizando el framework Flask, donde se desplegaron los algoritmos y modelos desarrollados. Además, este servidor puede almacenar registros cinemáticos de pacientes con diagnósticos confirmados en una base de datos PostgreSQL. La ampliación de esta base de datos mejorará los modelos de Machine Learning y reforzará las capacidades de la herramienta. El proceso de desarrollo y funcionamiento de la herramienta resultante está registrado como *TremorSoft*.

# Acknowledgments

I want to express my deepest gratitude to all the people who made significant contributions to the completion of this doctoral thesis:

- To Professor **Antonio J. Sánchez Egea** and Professor **Hernán A. González Rojas**, my thesis advisors, for their guidance, support, and valuable contributions throughout this research process.
- To my mother, **Myriam**, for her unwavering love and unconditional support and for being my constant source of inspiration.
- To my father, **Bernardo**, for the values and teachings he instilled in me, which continue to guide me in my academic and personal journey.
- To my siblings, **Iriana**, **Bernardo**, **Felipe**, and **Natalia**, for their affection and understanding throughout this journey.
- To my best friend, **Camilo**, for his unwavering support, wise advice, and shared moments of joy.
- To my partner, **Alejandra**, for her love, patience, and constant support throughout this entire process.
- To the **Universitat Politècnica De Catalunya** and the **Department of Mechanical Engineering** for allowing me to conduct this research.

Your support and encouragement were instrumental in the completion of this thesis. Thank you for being by my side on this journey.

**Julián David Loaiza Duque**

September 27, 2023

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

This work is situated within the challenges of the Horizon H2020 program [1], specifically Health, Demographic Change, and Well-being, as well as in the Health Cluster of Global Challenges and European Industrial Competitiveness [2] for 2021-2027. Its aim is to promote projects that improve the quality of life for elderly individuals and reinforce healthcare and assistance systems. Non-communicable diseases (NCDs) are a significant public health concern, accounting for approximately 71% of annual global deaths [3]. The onset of NCDs is influenced by various factors such as genetics, environment, and behavior. Unhealthy eating habits, smoking, alcohol abuse, and a sedentary lifestyle are known to increase the risk of developing and dying from these diseases, particularly in individuals between ages 30 and 69. It is crucial to prioritize efforts to prevent and manage NCDs to reduce their impact on global health. High mortality rates are primarily caused by the challenges of early detection and inadequate treatments to mitigate their impact. Thus, it is critical to conduct early evaluations, diagnoses, treatments, and palliative care for addressing NCDs. This study focuses on Movement Disorders, specifically Parkinson's Disease and Essential Tremor, which are among the most prevalent tremor syndromes globally [4, 5].

The global prevalence rate of Parkinson's Disease among elderly individuals is estimated to be 1. 6%[6], and that of essential tremor is 4. 6%[7], or approximately 474 million people worldwide. Distinguishing between these two conditions in the early stages, particularly in Parkinson's Disease patients with no family history, can be uncertain, as noted by [4, 8–10]. The risk of misdiagnosis is considerable; approximately 40% of patients with Essential Tremor receive an inaccurate diagnosis [11], with Parkinson's Disease being the most common misdiagnosis [12]. Approximately 47 of Parkinson's Disease diagnoses are incorrect in primary care settings, while about 25% are incorrect when conducted by specialists not experienced in specific movement disorders. For specialists in movement disorders, between 6% and 8% of cases result in incorrect diagnoses [13, 14].

Resting tremors are typically linked to Parkinson's Disease, whereas postural or kinetic tremors are associated with Essential Tremor [5]. Nonetheless, some individuals with Parkinson's Disease may experience postural tremor [5], and some individuals with Essential Tremor may have resting tremor as the disease progresses [15, 16]. Early diagnosis is essential to guarantee appropriate patient treatment and prevent harmful side effects [4, 5, 17]. While medical evaluations remain the preferred diagnostic method for Parkinson's and Essential Tremor [18], using dopamine transporter single-photon emission computed tomography (DaT-SPECT) with (123I)ioflupane is the most reliable complementary technique for Parkinson's diagnosis, with a sensitivity of up to 99.4% [11, 19]. However, this test is invasive and requires patient compatibility with the radiopharmaceutical tracer, which may hinder its applicability [20]. Moreover, this test is costly, and access to this technology is scarce or non-existent in several countries, especially developing regions. Addressing these challenges highlights the need for economical and user-friendly technology to facilitate the diagnosis of movement disorders.

## 1.1 Movement Disorders

Movement Disorders are neurological conditions that lead to abnormal movements or tremors, which can manifest either voluntarily or involuntarily [21]. As defined by the Task Force [22], a tremor is a rhythmic, oscillating, involuntary movement of a body part. This manifestation is most notable in fingers, hands, legs, head, and voice but is not observed during sleep [23]. When the limbs and head lack support, they display a minor tremor known as a physiological tremor. This type of tremor usually decreases amplitude and only affects fine motor control [22, 24]. Unless aggravated by fatigue or anxiety, physiological tremors are usually not visible or symptomatic. In contrast, pathological tremors are more pronounced and persistent [22].

The tremor classification is based on clinical characteristics and activation conditions according to Table 1.1[25]. The Task Force suggested a classification of tremors based on two axes: clinical features (Axis 1) and etiology (Axis 2)[22, 25]. This two-axis approach expedites the collection of clinically relevant data from patients with tremors, serving as a diagnostic and research tool [22].

The clinical characteristics of tremors include various aspects of medical history such as the age of onset, family history, temporal course, exposure to drugs and toxins, as well as the characteristics of the tremor, comprising body distribution, activation conditions, and frequency, along with associated systemic and neurological signs. For certain types of tremors, additional characterizations can be obtained through laboratory analyses, for instance, frequency recordings for orthostatic tremors and structural imaging for lesion localization. On the other hand, analyzing serum and

Table 1.1: Tremor Classification Based on Clinical Characteristics and Activation Conditions

| | |
|---|---|
| Rest | When the limbs are fully supported against gravity |
| Action | During different types of movements |
| Postural | While holding a limb or body part in one position, against gravity |
| Kinetic | With directed voluntary movement |
| Intent / Terminal | While moving the limb towards a target |
| Isometric | While contracting the muscle without an observable movement |
| Task-specific tremor | During the execution of specialized tasks (writing, playing a musical instrument, etc.) |

tissue, along with biomarkers, can provide supplementary information for identifying Axis 2 etiologies. These causes could be genetic, acquired, or idiopathic (either familial or sporadic). Etiological factors comprise neurodegenerative diseases, chromosomal aneuploidy, mitochondrial genetic disorders, infectious and other inflammatory diseases, endocrine and metabolic disorders, neuropathies, spinal muscular atrophies, toxins, and drugs. The functional tests for classifying tremor syndromes along Axis 1 (Tests 1, 2, and 3) and Axis 2 (Tests 2, 3, and 4) are presented in Table 1.2 [26].

Table 1.2: Diagnostic Tests for Tremor Classification

| | |
|---|---|
| Electrophysiological testing | Surface EMG to record the tremor's presence, measure tremor frequency, and evaluate the morphology and rhythm of EMG. |
| | Fourier analysis of accelerometric and EMG recordings with and without weight in hand to identify the mechanical reflex and central neurogenic tremors. |
| | Coherence and Fourier analysis of EMG recordings of multiple limbs to diagnose primary orthostatic tremor. |
| Structural imaging | MRI and CT for the detection of lesions, metabolic disorders, etc. |
| Receptor imaging | Dopamine and serotonin transporter imaging for disorders or deficiency syndromes. |
| Serum and tissue markers | Metabolic blood tests, infection tests, genetic tests, etc. |

### 1.1.1 Parkinson's Disease

Parkinson's disease is a neurodegenerative process that typically begins in adulthood and is the second most common neurodegenerative disorder after Alzheimer's dementia [27, 28]. The disease is characterized by the progressive loss of dopaminergic neurons in the substantia nigra pars compacta (SNc) of the mesencephalic region, as well as the presence of intracellular inclusions known as Lewy bodies [29–31]. The presence of Lewy bodies is the primary distinguishing feature of

Parkinson's disease [32, 33], although this disease also affects several other neurotransmitters such as acetylcholine, serotonin, and noradrenaline [34]. This neurodegeneration results in the denervation of dopaminergic SNc projections to the striatum, which disrupts the normal physiology of the basal ganglia and leads to remarkable manifestations of the disease [12, 31, 35].

Although there have been significant advancements in both structural and functional brain imaging techniques, Parkinson's Disease diagnosis still primarily relies on clinical observations [36, 37]. In this context, misdiagnosis rates can reach up to 47% [13, 14], and an autopsy is required to confirm the disease definitively [38]. However, the UK Brain Bank has introduced the most widely accepted clinical criteria, incorporating four cardinal signs: bradykinesia, resting tremor, rigidity, and postural instability [8, 39].

Table 1.3 presents the levels of confidence in diagnosing Parkinson's Disease, which include possible, probable, and definite diagnoses. To make a possible diagnosis, the patient must have at least two of the four cardinal signs, with at least one of them being tremor or bradykinesia. Furthermore, there must be no features indicating an alternative diagnosis, and the patient must not exhibit a positive response to dopaminergic medications. To reach a probable diagnosis of Parkinson's, at least three of the four signs must be present, and an alternative diagnosis must be eliminated. Additionally, a verified and continued response to dopaminergic medications must be documented. A definite diagnosis of Parkinson's Disease necessitates meeting all the criteria for a possible diagnosis and receiving histopathological confirmation. After diagnosis, patients are classified based on their stage of the Hoehn and Yahr Scale (Table 1.4), a tool used to assess disease progression and severity and evaluate treatment efficacy. It is important to note that the scale solely focuses on motor symptoms [31].

### 1.1.2 Essential Tremor

Essential Tremor is a prevalent movement disorder among individuals over 60 years old, with an estimated prevalence exceeding 5-6% [16, 24]. Essential Tremor onset primarily occurs in the second and sixth decades of life, where 91% of younger subjects display a family history of tremor [25, 40]. Although Essential Tremor is a common disorder, it has been extensively researched due to its impact on the quality of life of those affected. The most characteristic manifestation of Essential Tremor is a postural or kinetic tremor, with frequencies typically ranging from 5 to 8 Hz [40, 41]. Although initially regarded as a benign monosymptomatic disorder, recent research has revealed its complexity and associated neurodegeneration.

Throughout history, Essential Tremor has been viewed as a focal clinical condition defined by

Table 1.3: UK Parkinsons Disease Society Brain Bank Diagnostic Criteria

| STEP 1. Diagnosis of Parkinsonian syndrome |
|---|
| Bradykinesia (slowness of initiation of voluntary movement with progressive reduction in speed and amplitude of repetitive actions). |
| And at least one of the following: |
| a. Muscular rigidity |
| b. 46 Hz rest tremor |
| c. Postural instability is not caused by primary visual, vestibular, cerebellar, or proprioceptive dysfunction. |
| **STEP 2. Exclusion criteria for Parkinsons disease** |
| History of repeated strokes with the stepwise progression of Parkinsonian features. |
| History of repeated head injury. |
| History of definite encephalitis. |
| Oculogyric crises. |
| Neuroleptic treatment at the onset of symptoms. |
| More than one affected relative. |
| Sustained remission. |
| Strictly unilateral features after three years. |
| Supranuclear gaze palsy. |
| Cerebellar signs. |
| Early severe autonomic involvement. |
| Early severe dementia with disturbances of memory, language, and praxis. |
| Babinski sign. |
| Presence of a cerebral tumor or communicating hydrocephalus on CT scan. |
| Negative response to large doses of levodopa (if malabsorption excluded). |
| MPTP (1-methyl-4-phenyl-1,2,3,6-tetrahydropyridine) exposure. |
| **STEP 3. Supportive prospective positive criteria for Parkinsons disease. Three or more required for the diagnosis of definite Parkinsons disease** |
| Unilateral onset. |
| Rest tremor present. |
| Progressive disorder. |
| Persistent asymmetry affects the side of onset most. |
| Excellent response (70100%) to levodopa. |
| Severe levodopa-induced chorea. |
| Levodopa response for five years or more. |
| The clinical course of 10 years or more. |

Table 1.4: Hoehn and Yahr Scale for Parkinson's Disease Staging

| | |
|---|---|
| Stage 0 | No signs of disease |
| Stage 1 | Symptoms on one side only (unilateral) |
| Stage 1.5 | Symptoms unilateral and also involving the neck and spine |
| Stage 2 | Symptoms on both sides but no impairment of balance |
| Stage 2.5 | Mild symptoms on both sides, with recovery when the pull test is given (the doctor stands behind the person and asks them to maintain their balance when pulled backward) |
| Stage 3 | Balance impairment, mild to moderate disease, physically independent |
| Stage 4 | Severe disability, but still able to walk or stand unassisted |
| Stage 5 | Needing a wheelchair or bedridden unless assisted. |

tremors, characterized by the absence of other neurological signs [40]. However, recent research has revealed that Essential Tremor is not solely an isolated movement disorder, but rather, it is associated with considerable non-motor symptoms such as depression, anxiety, and mild cognitive deficits within areas like attention and memory [26, 42]. The evolving perception of Essential Tremor as a multifaceted and intricate disorder has been established by empirical research involving clinical observation, neuroimaging, and pathophysiological studies. These investigations have successfully identified notable alterations in both the structure and function of specific brain regions [42].

One emphasized aspect of Essential Tremor research is the increasing recognition of its clinical and etiological heterogeneity [26, 43]. Essential Tremor is a non-uniform entity encompassing various clinical presentations and potential underlying causes. This partially results from the diversity of motor and non-motor symptoms observed in Essential Tremor patients. The variations in symptoms and potential causes have challenged prior understandings of a singular medical condition, resulting in a more intricate comprehension of the disorder.

The diagnosis of Essential Tremor typically depends on clinical evaluation and neurological history, given the absence of precise biological indicators or dependable diagnostic tests [44]. However, the International Parkinson and Movement Disorder Society published a consensus statement in 2017 presenting diagnostic criteria and a revised classification of Essential Tremor [22]. The consensus recognizes two main categories of Essential Tremor (refer to Table 1.5): Simple Essential Tremor and Complex Essential Tremor (or ET-Plus), reflecting the diversity of clinical presentations and associated symptoms [22, 45].

In addition to setting forth criteria for classification and diagnosis, the 2017 consensus furthermore presents significant exclusion criteria (Table 1.6) for diagnosing Essential Tremor [22]. These exclusions are essential for eliminating other ailments manifesting with symptoms that resemble

Table 1.5: Inclusion Criteria for Essential Tremor Classification

| | |
|---|---|
| Essential Tremor | Isolated tremor syndrome of bilateral upper limb action tremor<br>At least three years duration<br>With or without tremor in other locations (e.g., head, voice, or lower limbs)<br>Absence of other neurological signs, such as dystonia, ataxia, or parkinsonism. |
| Essential Tremor plus | Tremor with the characteristics of Essential Tremor<br>Additional neurological signs of uncertain significance such as impaired tandem gait, questionable dystonic posturing, memory impairment, or other mild neurologic signs of unknown significance that do not suffice to make an additional syndrome classification or diagnosis. Essential Tremor with tremor at rest should be classified here. |

Essential Tremor but warrant a different therapeutic strategy. This exclusionary approach is instrumental in enhancing diagnostic precision and identifying the appropriate subjects for ET-focused research and treatment.

Table 1.6: Exclusion Criteria for Essential Tremor Classification

| |
|---|
| Isolated focal tremors (voice, head) |
| Orthostatic tremor with a frequency >12 Hz |
| Task- and position-specific tremors |
| Sudden onset and step-wise deterioration |

This section discusses movement disorders, specifically Parkinson's Disease and Essential Tremor. These disorders pose substantial challenges to health and well-being due to their widespread occurrence and the negative impact on affected individuals' quality of life. The classification criteria and highlighted diagnostic challenges emphasize the intricacy of accurately identifying and differentiating these conditions, even for specialists [46]. The section below explores the concepts and methodologies of machine learning and its potential to bolster early detection, treatment, and patient care for individuals with Parkinson's Disease and Essential Tremor. This contribution strives to enhance healthcare systems and improve the overall well-being of affected individuals.

## 1.2 Machine Learning Algorithms

Machine learning, a subfield of artificial intelligence, entails designing algorithms and models that can learn from data and generate predictions and decisions without explicit programming [47]. It has gained wide popularity in recent years owing to its versatile applications across various domains, including image and speech recognition, natural language processing, recommendation systems, and autonomous vehicles [48, 49]. This section addresses the working principles of machine learning

algorithms employed in developing classification models [50]. These models receive multiple input variables, particularly the kinematic features of the signal, and predict whether the subject under study can be classified as having physiological or pathological tremors and subsequently distinguish between Parkinson's pathology and Essential Tremor.

Different models have their own benefits and drawbacks, and choosing the most appropriate model for each study is essential. Table 1.7 provides an overview of the four main types of algorithm families and some of the most frequently implemented algorithms within each family. This research concentrates on the family of supervised algorithms, which can be either based on regression or classification, as described in [48].

Table 1.7: Classification of Machine Learning families and some lerated algorithms.

| Machine Learning Algorithms | | | | | | |
|---|---|---|---|---|---|---|
| Supervised | | Unsupervised | | Semi-Supervised | Reinforcement | |
| Regression | Classification | Clustering | Association | | Model-Free | Model-Based |
| Linear Regression | Naïve Bayes | k-Means | Apriori | Heuristic | Q-learning | Markov Decision |
| Logistic Regression | Logistic Regression | Mean Shift | Eclat | Graph-based Method | Monte Carlo | Imagination-Augmented Agents |
| Decision Tree | Support Vector Machine | Gaussian Mixture | FP Growth | Low Density Separation | Policy Optimization | Model-Based Value Expansion |
| Neural Network | K-Nearest Neighbor | DBScan | | | | |
| Lasso Regression | Random Forest | | | | | |
| Support Vector Machine | Gradient Boosting | | | | | |
| Polynomial Regression | Decision Tree | | | | | |

### 1.2.1 Supervised Machine Learning

Supervised machine learning is a ubiquitous algorithm in machine learning. Its principal aim is to map input data to output labels using a training set of known input-output pairs [50]. This framework is highly adaptable and potent in addressing classification and regression issues in the machine-learning sphere. The premise is that input data is labeled, and the objective is to develop a mapping from input to output variable that can apply to unobserved data. Each algorithm has its own set of suppositions, limitations, and advantages, and the algorithmic selection should be tailored to the problem and the available data.

In supervised machine learning, a collection of input-output pairs (training examples) are given to the algorithm, which learns to map inputs to outputs by recognizing patterns in the data. After training, the algorithm can use the learned mapping to make predictions or decisions regarding new data. The objective is to construct a function that can predict output values for fresh inputs with precision. This function is usually exemplified by a model containing coefficients and weights learned from the training data. The evaluation of the model quality is carried out on an autonomous set of test data. Performance evaluation metrics for supervised machine learning models include

accuracy, precision, recall, F1 score, and area under the receiver operating characteristic curve (AUC-ROC) [51, 52].

Supervised machine learning encompasses two subcategories: regression and classification. In regression, the machine learning model predicts a continuous output variable based on input data. In contrast, the output variable in classification is discrete, and the objective is to acquire a function that assigns inputs to various predefined classes. Figure 1.1 illustrates the distinction between these subcategories.

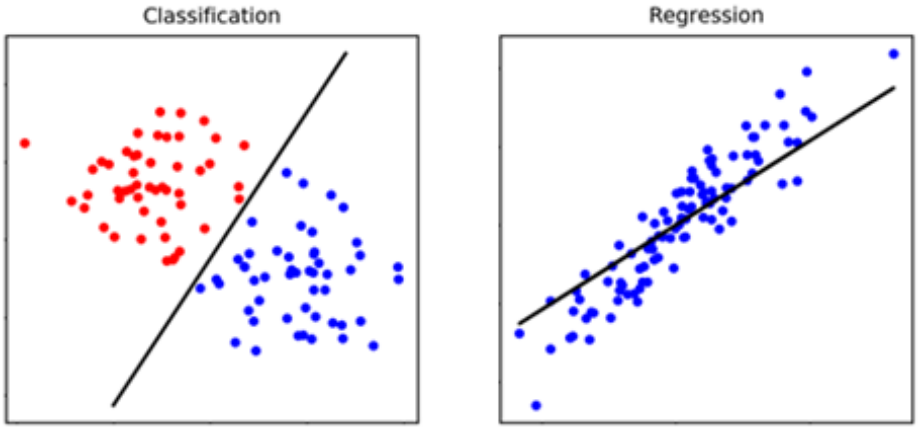



Figure 1.1: Comparison of regression and classification models in the context of supervised Machine Learning.

In the field of Supervised machine learning, there are several widely used algorithms, each with its strengths and limitations. Some of the most common algorithms are described below.

### Linear Regression

Linear regression is a supervised learning technique that establishes a relationship between one or more independent variables and a continuous dependent variable through a linear function [53]. The primary objective of linear regression is to identify coefficients that minimize the sum of squared errors between forecasted and actual values of the dependent variable. Given a training dataset $D$ consisting of $N$ examples and a feature set $X$, Linear Regression can be represented as the process of constructing and predicting as follows:

1. Define the linear model:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p + \epsilon \tag{1.1}$$

where $y$ is the dependent variable, $\beta_0$ is the intercept term, $\beta_i$ are the coefficients associated with the independent variables, $x_i$ are the independent variables, $p$ is the number of

independent variables, and $\epsilon$ is the error term.

2. Adjust the coefficients $\beta_i$ to minimize the sum of squared errors between the predicted and actual values of the dependent variable. This is done using the method of least squares. The process of fitting the coefficients $\beta_i$ involves solving the normal equations:

$$\beta = (X^T X)^{-1} X^T y \tag{1.2}$$

where $X$ is the matrix of independent variables, $y$ is the vector of dependent variable values, and $\beta$ is the coefficient vector.

3. For a new input $x$, compute the prediction $y$ using the fitted linear model:

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p \tag{1.3}$$

Linear regression is a straightforward and frequently employed algorithm in finance, economics, and social sciences. Nevertheless, it has constraints when confronted with non-linear variable relationships or the existence of outliers or missing values within the data [48]. Despite its simplicity, this algorithm is a prevalent method based on its interpretability and adaptability in varied domains.

### Logistic Regression

Logistic Regression is a binary classification algorithm that models the relationship between a binary dependent variable and one or more independent variables. The primary objective of Logistic Regression is to identify a sigmoid curve that optimally distinguishes the two classes [54, 55]. Given a set of training data, $D$, with $N$ examples, and a set of features, $X$, and assuming binary classification with labels $y \in \{0, 1\}$, the process of constructing and predicting with Logistic Regression can be represented as follows:

1. Define the logistic function (sigmoid):

$$p(y = 1|x) = \frac{1}{1 + \exp(-z)} \tag{1.4}$$

where $z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p$ is the linear predictor, and $\beta_i$ are the coefficients associated with the independent variables.

2. Adjust the coefficients $\beta_i$ to maximize the probability of observed labels in the training set. This is done by finding the values that maximize the likelihood function:

$$L(\beta) = \prod_{i=1}^{N} p(y_i|x_i; \beta) \tag{1.5}$$

3. For a new input $x$:

   (a) Calculate the probability of the positive class as $p(y = 1|x)$.

   (b) Predict the positive class label if $p(y = 1|x) > 0.5$, otherwise predict the negative class label.

This technique is highly effective and has a range of applications, including credit scoring, medical diagnosis, and spam filtering. However, it assumes a linear relationship exists between independent variables and the log-odds of the positive class, which may not hold true in certain contexts. Although its assumptions limit its applicability in some instances, Support Vector Machines remain popular due to their simplicity and effectiveness in various applications.

### Support Vector Machines

Support Vector Machines are supervised learning algorithms for classification and regression tasks [56, 57]. The primary objective of Support Vector Machines (SVM) is to determine the optimal hyperplane that maximizes the separation between classes in the feature space [58]. This hyperplane seeks to separate classes with the largest margin possible for classification cases. The process of constructing and predicting with SVM assumes binary classification with labels $y \in \{-1, 1\}$, and it requires a training set $D$ with $N$ examples and a feature set $X$.

1. Find the hyperplane that maximizes the margin:

$$w^T x + b = 0 \tag{1.6}$$

   where $w$ is the normal vector to the hyperplane, and $b$ is the bias term.

2. Define the decision boundaries:

$$w^T x + b \geq 1 \quad \text{if} \quad y = 1 \tag{1.7}$$

$$w^T x + b \leq -1 \quad \text{if} \quad y = -1 \tag{1.8}$$

3. Solve the optimization problem to find $w$ and $b$ that maximize the margin while satisfying the constraints:

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}||w||^2 \\ \text{subject to} \quad & y_i(w^T x_i + b) \geq 1, \quad \text{for} \quad i = 1, ..., N \end{aligned} \tag{1.9}$$

4. For a new input $x$, calculate the prediction as the sign of the hyperplane:

$$F(x) = \text{sign}(w^T x + b) \tag{1.10}$$

SVMs are effective in high-dimensional spaces and are applied in various fields, such as text classification, medical diagnosis, and fraud detection. Yet, selecting the appropriate kernel and parameter configuration may pose challenges and impact model performance. Linear, polynomial, and radial basis function (RBF) kernels remain the typical kernel types employed.

### Naive Bayes

The Naive Bayes algorithm is a supervised machine learning technique primarily utilized for classification [59]. It is founded on the principles of Bayes' theorem and conditional independence, resulting in the term "naive". Although simple, Naive Bayes has demonstrated efficacy in various applications, notably in natural language processing and document classification. Given a dataset $D$ consisting of $N$ examples and a feature set $X$, the Naive Bayes algorithm can be used to construct and predict.

1. Calculate the prior probabilities of each class:

$$P(Y = c) = \frac{\text{Number of examples with class } c}{N} \tag{1.11}$$

2. Calculate the conditional probabilities of each feature given a class:

$$P(X_j = x_j^{(i)} | Y = c) = \frac{\text{Number of examples with } X_j = x_j^{(i)} \text{ and class } c}{\text{Number of examples with class } c} \tag{1.12}$$

   where $X_j$ is the $j$-th feature and $x_j^{(i)}$ is the value of the $j$-th feature in the $i$-th example.

3. For a new input $x$, calculate the posterior probabilities for each class using Bayes' theorem:

$$P(Y = c|X) = \frac{P(Y = c) \cdot P(X|Y = c)}{P(X)} \tag{1.13}$$

   where $P(X|Y = c)$ is estimated by multiplying the conditional probabilities of each feature given the class.

4. Assign the class with the highest posterior probability as the prediction.

The algorithm assumes conditional independence, implying that features are independent of each other given the class. However, if this assumption does not hold in the problem domain, then Naive Bayes may lead to lower prediction accuracy. The Naive Bayes algorithm operates at an impressive speed, particularly when handling massive datasets, and its efficacy can be remarkably impressive given its basic design.

### k-Nearest Neighbors

The k-Nearest Neighbors (k-NN) algorithm is a machine learning technique for classification and regression tasks proposed by Altman [60] and Freund [61]. It operates based on the concept that comparable examples usually possess the same labels or values. The k-NN method makes predictions by selecting the k-nearest examples from the training set, then determining an output based on most of their labels (in Classification) or by averaging their values (in Regression). Given a set of training data $D$ containing $N$ examples and a feature set $X$, the k-NN method for constructing and predicting can be summarized as follows:

1. Define a distance metric, such as Euclidean distance, to measure similarity between examples.
2. For a new input $x$:
   (a) Calculate the distance between $x$ and each training example $x_i$ using the distance metric.
   (b) Select the $k$ nearest training examples to $x$.

   In the case of Classification:

   (a) Count the frequencies of labels of the $k$ nearest neighbors.
   (b) Assign the most common label as the prediction.

   In the case of Regression:

   (a) Calculate the average output variable values of the $k$ nearest neighbors.
   (b) Use the average as the prediction.

The user must choose a hyperparameter value $k$. Opting for a small $k$ value may lead to a model that is highly sensitive to noise, while a larger value can mitigate noise effects but reduce details at decision boundaries. The k-NN algorithm is straightforward and comprehensible, but its efficacy hinges on selecting the distance metric and the value assigned to $k$. Moreover, it can incur significant computational costs when applied to large datasets. Nevertheless, its simplicity and adaptability make it a valuable tool for classification and regression.

### Decision Trees

Decision trees are supervised machine learning algorithms used for classification and regression tasks [62, 63]. These algorithms create a tree model with internal nodes representing features,

branches indicating decision rules, and leaf nodes displaying outcomes. The process of constructing the tree involves recursively dividing the dataset into smaller subsets, using selected features to maximize the purity of each class in the leaf nodes. Mathematically, the construction of a decision tree can be formulated as follows:

1. Given a training set $D$ with $N$ examples and a feature set $X$, the algorithm searches for the feature $X_j$ and threshold value $T$ that best separates the examples into two groups, $D_{left}$ and $D_{right}$.

2. An impurity metric, such as Gini impurity or information gain, is calculated to evaluate the quality of the split. Impurity is defined as:

$$\text{Impurity}(D) = 1 - \sum_{c=1}^{C} P(c|D)^2 \tag{1.14}$$

   where $C$ is the number of classes, and $P(c|D)$ is the proportion of class $c$ examples in set $D$.

3. The process is repeated recursively for subsets $D_{left}$ and $D_{right}$, generating additional internal nodes and splitting the data until a stopping criterion is met, such as a maximum depth or a minimum number of examples in a node.

4. For a new input $x$:

   (a) Traverse the tree following decision rules based on features.

   (b) Reach a leaf node and assign the corresponding label or value as the prediction.

Decision Trees are recognized for their interpretability and visualization abilities, which are highly valued in applications prioritizing model transparency. Nonetheless, deep and complicated trees pose a risk of overfitting. To address this issue, various techniques, including pruning, gradient boosting, and random forests, have been developed to improve the model's generalizability.

### Random Forests

Random Forests are an ensemble learning algorithm that combines multiple decision tree models to enhance prediction accuracy and robustness [64, 65]. This algorithm constructs a set of decision trees on random subsets of input variables and training data. The final prediction is obtained by averaging the predictions of individual Trees. The process of constructing and predicting with Random Forests can be represented as follows: A training dataset, comprising $N$ examples and $M$ features, is used alongside a set of $T$ decision trees.

1. For each tree $t$ in the forest ($t = 1, 2, ..., T$):

   (a) Select a subset of $N$ examples from the training set $D$ with replacement (bootstrap).

   (b) Select a random subset of $m$ features from the total $M$ features.

   (c) Build a decision tree using the selected example and feature subset.

2. For a new input $x$:

   (a) For each tree $t$ in the forest, calculate the tree's prediction $h_t(x)$.

   (b) Calculate the final forest prediction by averaging the tree predictions:

$$F(x) = \frac{1}{T} \sum_{t=1}^{T} h_t(x) \tag{1.15}$$

Random Forests efficiently avert overfitting and enhance accuracy by minimizing the intrinsic variability of individual trees. Random Forests introduce diversity into the trees by using random subsets of examples and features, making the forest more resilient to noisy data and irrelevant variables. Additionally, averaging across multiple trees mitigates the impact of individual errors, enhancing generalization. Although Random Forests are powerful and user-friendly, constructing multiple trees can lead to computational expense. However, more efficient alternatives exist, such as the Extremely Randomized Trees (ExtraTrees) algorithm. This algorithm selects random split points rather than searching for the best split points for each feature. In summary, Random Forests are a valuable tool in machine learning, capable of handling complex and noisy datasets and improving accuracy and robustness. Their ability to average multiple trees and reduce overfitting makes Random Forests a popular tool for various applications.

### Gradient Boosting Machines

Gradient Boosting Machines (GBM) are machine learning algorithms that combine multiple weak models to form a strong model [66]. Typically, GBM employs shallow decision trees as the weak models. The fundamental concept behind GBM is to iteratively fit new weak models to the residuals of previous models to enhance the final model's prediction. Mathematically, the process of constructing and predicting with GBM can be expressed as follows:

1. Given a training set $D$ with $N$ examples and a feature set $X$, we start with an initial weak model $F_0$, which can be a simple estimate like the mean of output values.

2. In each iteration $m$, we fit a new weak model $h_m$ to the residuals of the previous model $F_{m-1}$:

$$h_m = \arg\min_h \sum_{i=1}^{N} L(y_i, F_{m-1}(x_i) + h(x_i)) \tag{1.16}$$

Where $L$ is a loss function that measures the discrepancy between predictions and actual values $(y_i)$.

3. We update the final model $F_m$ by adding the adjusted weak model weighted by a learning rate $\eta$:

$$F_m(x) = F_{m-1}(x) + \eta \cdot h_m(x) \tag{1.17}$$

4. For a new input $x$, we use the final model $F_m(x)$ to make predictions:

$$Prediction for\, x: \quad \hat{y} = F_m(x) \tag{1.18}$$

GBM has the ability to capture intricate and nonlinear connections between input and output variables. Variations of GBM, including XGBoost, LightGBM, and CatBoost, integrate optimizations and extra features that improve overall performance and training efficiency [67]. It is essential to note that choosing the appropriate learning rate and hyperparameters can significantly impact model performance. Moreover, GBM may be more susceptible to overfitting than other algorithms due to its sequential approach to fitting residuals.

### Neural Networks

Neural networks are models inspired by the structure and function of the human brain [68]. They are powerful and flexible algorithms capable of learning complex patterns in data. A neural network comprises interconnected nodes, known as artificial neurons, arranged in layered configurations. Every node linearly combines inputs and applies a non-linear activation function. Neural networks can have several hidden layers between input and output layers, known as deep neural networks or "deep learning." Given a dataset consisting of $N$ examples and a feature set $X$, the process of constructing and predicting with neural networks can be depicted as follows:

1. Define the forward propagation operation:

$$z^{(l)} = W^{(l)} a^{(l-1)} + b^{(l)} \tag{1.19}$$

$$a^{(l)} = f(z^{(l)}) \tag{1.20}$$

where $l$ denotes the layer, $W^{(l)}$ is the weight matrix, $a^{(l)}$ is the output of layer $l$, $b^{(l)}$ is the bias vector, and $f(\cdot)$ is the nonlinear activation function.

2. Adjust the weights and biases to minimize errors between predictions and actual values. This is done using optimization algorithms like gradient descent. The process involves calculating partial derivatives of the error concerning the network's parameters and updating values in the direction that minimizes the error.

3. For a new input $x$:

   (a) Calculate the output of the last layer using the forward propagation operation.

   (b) Predict the label or output value based on the Neural Network's configuration.

Neural Networks can detect intricate relationships and are implemented in image recognition, natural language processing, and gaming, among other tasks. Nevertheless, they can demand substantial computational resources and necessitate extensive training datasets to evade overfitting [49]. Interpreting the decisions made by neural networks can be challenging due to their complex and opaque internal operations. Despite their data requirements and complexity, the effectiveness of neural networks across various applications makes them valuable for solving complex problems.

Supervised machine learning algorithms are powerful tools for addressing various classification and regression problems. From linear regression and logistic regression to decision trees, support vector machines, and neural networks, each algorithm has its own advantages and disadvantages regarding complexity, interpretation, and performance in varied situations. The selection of an appropriate algorithm largely depends on the specific problem, the availability of data, and the required level of accuracy. It is often necessary to test multiple algorithms and fine-tune their hyperparameters to achieve the best performance for a given task. As machine learning advances, new algorithms and approaches will likely emerge, leading to further possibilities for applications and increased accuracy.

## 1.3 Motivation

The reliable and early diagnosis and monitoring of Parkinson's Disease is currently a focus of research [69, 70]. Optical motion detection systems [71] are one of the proposed and implemented non-invasive techniques that extract kinematic information from patients, aiding in clinical monitoring and diagnosis. Nevertheless, their practicality is frequently limited by the necessity for expensive and sophisticated equipment, a fixed camera range that constrains the workspace for recording movement, and the necessary training for operation.

Inertial measurement units (IMUs), such as accelerometers, have been integrated into these

systems to enhance their capabilities recently [72]. The analysis of accelerometer and gyroscope data is a crucial area of investigation in biomechanics, as it allows for recording movement information through wearable devices [73–75]. Researchers are actively studying the use of these devices in movement disorders [76], resulting in the publication of numerous papers on the subject. Many of these studies concentrate on the diagnosis of such disorders. For example, Uchida et al. [15] used a triaxial accelerometer to measure the severity and frequency of hand tremors in patients with Essential Tremor and Parkinson's disease during various tasks. They found that resting tremors were reduced during walking in Essential Tremor patients and increased in Parkinson's. Similarly, Wile et al. [41] classified patients by analyzing their Mean Harmonic Power using a smartwatch accelerometer. Locatelli et al. [5] discovered that a smartwatch device was more effective than an analog accelerometer in distinguishing between tremor subjects. They used a wearable sensor to record hand tremors during various tasks, differentiating between patients with tremors. Notably, they found that the frequency domain was more indicative of Parkinson's Disease during resting tasks, while Essential Tremor subjects exhibited more distinct data during postural and kinetic tasks.

In recent years, researchers such as Bernhard et al. [77] have investigated gait and balance deficits using wearables placed on the lower back and ankle, claiming that such devices could monitor the progression of movement disorders and response to treatment. Additionally, Varghese et al. [78] have created the Smart Device System, which discovers fresh phenotypical biomarkers in individuals with movement disorders, training AI models to predict such conditions. Machine learning algorithms have been incorporated in these studies, enabling a thorough kinematic analysis of movement disorders [79]. This assists in the identification and differentiation of tremor conditions. Surangsrirat et al. [10] utilized machine learning algorithms to categorize patients with movement disorders based on angular velocity fluctuations detected by a 6-degree-of-freedom (DOF) inertial measurement unit. Meanwhile, Raza et al. [80] assessed the efficacy of their machine learning models in discriminating between Parkinson's Disease and Essential Tremor subjects, compared to early diagnoses by medical specialists. Kramer et al. [81] combined electromyography and accelerometry signals, using Wavelet Coherence Analysis to differentiate between tremor types, which yielded better results than standard coherence analysis.

In a recent study, Ricci et al. [82] employed a network of wearable sensors to measure kinematic features during a range of tasks performed by Parkinson's patients and healthy subjects. Machine learning algorithms were employed to attain a 95% accuracy in distinguishing between the two groups. The evolution of wearable technologies has opened the path for mobile apps in the medical industry to diagnose, assess, analyze, and monitor movement disorders [83, 84]. However, only a few

systems have been fully integrated as ready-to-use applications. Researchers have primarily concentrated on providing objective data on disease progression and monitoring its development [85]. However, currently, no low-cost, easily accessible, and user-friendly solutions exist which can accurately, securely, and consistently differentiate between various movement disorders [86]. To address this issue, LaMoyne et al.[87] developed an iPhone app to characterize hand tremors in Parkinson's patients using the built-in triaxial accelerometer. The application recorded acceleration data and enabled email transmission for subsequent analysis. According to Daneault et al. [88], smartphones are useful for evaluating anomalous motor variables, and they developed an app that characterizes tremor acceleration. Using a support vector machine model, Woods et al. [24] formulated an offline app with a smartphone accelerometer to differentiate patients with Parkinson's Disease and Essential Tremor with an accuracy of 96%. The AWARE framework, an open-access application created by Ferreira et al. [89], facilitates the collection of data from inertial sensors in smartphones. Kostikis et al. [90] developed a web application to collect and process data from tremor patients and healthy subjects, classified using machine learning models. The TREMOR12 app for Apple devices, developed by Kubben et al. [84], quantifies tremors using accelerometers and gyroscopes. Kuosmanen et al. [91] additionally introduced the STOP mobile app, which quantifies tremor severity and medication efficacy in Parkinson's patients using accelerometer data.

Current research is focused on developing dependable, inexpensive decision-support tools for distinguishing between various movement disorders. An appealing solution involves the integration of wearables with machine learning algorithms; however, there is a shortage of thorough diagnostic applications [86]. The challenge is to create standardized, easy-to-use tools for distinguishing accurate tremor diseases, particularly in resource-limited environments. In previous studies [4, 92], methodologies have been proposed to perform differential classification of Parkinson's disease and essential tremor using inertial sensors from wearable devices, such as the built-in accelerometer of a mobile phone. Barrantes' approach identified specific features that distinguished hand tremors in patients with tremor disease.

The ongoing advancement of wearable technologies, smartphone applications, and machine learning techniques presents an opportunity to impact the diagnosis and monitoring of movement disorders significantly. One possibility to bridge the gap between the demand for advanced medical resources and the limitations of current diagnostic methods is to develop a comprehensive mobile application and web server that integrates wearable devices and machine learning algorithms. This study aims to contribute substantially to this vital field by presenting a non-invasive option to support diagnostic differentiation, particularly between Parkinson's disease and Essential Tremor.

## 1.4 Research topic

The primary aim of this project is to develop an affordable, non-intrusive, user-friendly quantitative technique that utilizes wearable technologies (e.g., smartphones and inertial sensors) and machine learning algorithms to facilitate differential diagnosis and remote monitoring of various movement disorders, particularly Parkinson's Disease and Essential Tremor. The research queries and specific goals of this research proposal can be succinctly outlined as follows:

### 1.4.1 Research questions

**Question 1:** Can hand tremors recorded with an inertial measurement unit be used to differentiate between Parkinson's disease and Essential Tremor? If yes, which sensor, gyroscope, or accelerometer offers the most specific information to differentiate these two tremor disorders?

**Question 2:** How do recording duration, recording system weight, and sampling rate affect the differential analysis of Parkinson's Disease and Essential Tremor patients? To what extent can machine learning classification methods enhance the differentiation between these movement disorders? What kinematic features are essential for implementing these classification methods?

**Question 3:** Can a mobile application be used to evaluate patients with abnormal tremor signs as a decision support tool? Is it feasible to estimate the severity of the tremor with the help of the mobile app after identifying the movement disorder?

## 1.5 Objectives

The specific objectives of this research can be summarized as follows:

- Design, develop, and implement data analysis algorithms based on the available data set.
- Apply feature selection techniques to evaluate and identify the kinematic features with the highest discriminative power.
- Train, test, and select the first set of classification models using the selected feature matrix.
- Plan and collaborate with specialists from the Department of Movement Disorders at various hospitals to register new data.

- Analyze the expanded database to determine the influence of different operational parameters on the performance of the classification models.

- Build the mobile application and the web server.

- Validate the operation of the final decision support tool with a small group of confirmed patients from the collaborating hospitals.

## 1.6 Research Impact and Excellence

### 1.6.1 Scientific and Technological Impact

The developed application offers a secure, efficient, and user-friendly classification of hand tremors. This is achieved through the integration of wearable devices with machine learning algorithms. The classification results are readily available during medical examinations, whether conducted on-site or remotely. This research differentiates itself from other similar studies in two key ways. First, the study simultaneously records accelerometer and gyroscope signals instead of only analyzing one data type like previous research. This approach combines features for model training, leading to more reliable models. Second, this research framework offers innovative techniques for analyzing hand tremor signals, presenting multiple novel features extracted from these signals. These attributes are noteworthy because of their multidimensionality, derived by utilizing data from postural and resting positions to define a distinctive characteristic. They possess a high discriminative capability, effectively identifying computational thresholds and hyperparameters that optimize the classification models of the mobile application and web server.

The clinical significance of this research is its ability to exceed current state-of-the-art methods for evaluating movement disorders, making it a valuable tool for distinguishing pathological tremors. This is particularly critical in cases where diagnosis is challenging, particularly in the early stages of a disease. The application's level of accuracy surpasses that of SPECT, demonstrating its reliability. This high level of reliability will aid neurologists in precisely evaluating and classifying the severity of movement disorders. The novel findings from this study will significantly contribute to improving the differential diagnosis of movement disorders. The primary objective of this application is to furnish neutral information to facilitate clinical decision-making and, most significantly, decrease the wait time to distribute fitting treatment to patients.

### 1.6.2 Socioeconomic Impact

The technology resulting from this research has potential benefits for three primary groups. The application aids the *treating physician* in the initial evaluation of a disease, particularly in cases where a diagnosis has not been made, in the early stages, or in complex cases that may necessitate supplementary tests. The application provides immediate information on upper limb tremors during medical evaluations, eliminating the need for more sophisticated and expensive techniques that often create bottlenecks in the healthcare system for doctors. Timely and accurate medical treatment can greatly impact a *Patient's health* by reducing complications and improving their quality of life. Finally, medical expenses for Parkinson's Disease patients can reach up to 17,000 euros annually in *clinics and hospitals in Spain* [93]. The proposed tool can potentially decrease healthcare costs by addressing incorrect or ineffective tremor treatments, expensive diagnostic technologies such as SPECT, and treatment expenses incurred prior to a definite diagnosis.

### 1.6.3 Technology Transfer Plan

The copyrighted tool, comprising a mobile application and web server, can be utilized by hospitals and non-governmental organizations (NGOs), such as the *Centro de Trastornos del Movimiento* (*CETRAM*, Chile), to benefit from its outcomes. The *Hospital Clínic Barcelona* and the *Hospital da Luz in Lisbon* are willing to incorporate this decision-support tool as an adjunct method in cases where physicians have uncertainties regarding the ultimate diagnosis. This know-how is of interest to both public institutions and private companies. A market niche exists wherein businesses focus on selling objective information applications for decision-making based on biomechanical variables, particularly via wearable devices. The project will delve into this sector to facilitate transferring the developed application to the market. Some companies that may be interested in this application are *ENGIDI SL* (located in Girona and specializing in biomedical sensors for safety enhancement in the workplace), *DyCare* (based in Barcelona and offering wearable solutions for rehabilitation), *MJN* (a Girona-based company that specializes in bioinstrumentation and machine learning), *Trimedica* (located in Madrid and specializing in biomedical sensors and applications), among others.

### 1.6.4 Dissemination and Outreach Activities

In addition to writing and presenting this study, several articles have been written, and the research results have been presented at international conferences. Furthermore, the initial operational version of the *TremorSoft* application has been created, and its entire source code has been made avail-

able online via *GitHub*. This availability facilitates sharing and collaboration with other medical professionals and researchers to enhance TremorSoft collectively. The long-term goals of TremorSoft involve continually improving the application through feedback from medical professionals, enhancing the tremor record database to train machine learning models better, and promoting the tool as a decision-support tool for movement disorder specialists in their daily clinical routines. Research collaborations have been conducted for dissemination activities with *Dr. Pedro Chaná* from the *CETRAM*, *Dr. Esther Catena Ruiz* from *Consorci Sanitari Alt Penedès-Garraf*, *Julia Barrero* from *Associació Catalana per al Parkinson (ACP)*, and *Joao Costa* and *Joaquim J Ferreira* from *Universidade de Lisboa*. These collaborations aim to foster the advancement of knowledge in the field. Technical abbreviations used will be explained throughout the text.

## Papers published

The publications resulting from this thesis have been featured in academic journals and international conference proceedings.

Julián D. Loaiza Duque, Andrés M. González-Vargas, Antonio J. Sánchez Egea, and Hernán A. González Rojas. Using Machine Learning and Accelerometry Data for Differential Diagnosis of Parkinsons Disease and Essential Tremor. In: *Communications in Computer and Information Science*. Vol. 1052. Springer, 2019, pp. 368378. ISBN: 9783030310189. DOI: 10.1007/978-3-030-31019-6_32 [94]

Abstract: Parkinsons disease and Essential Tremor are the most common tremor syndromes in the world. Currently, a specific Single Photon Emission Computed Tomography (123I-FP-CIT SPECT) has proven to be an effective tool for the diagnosis of these diseases (97% sensitivity and 100% specificity). However, this test is invasive and expensive, and not all countries can have a SPECT system for an accurate differential diagnosis of Parkinson's Disease patients. Clinical evaluation by a neurologist remains the gold standard for Parkinson diagnosis, although the accuracy of this protocol depends on the experience and expertise of the physician. Wearable devices have been found to be a potential tool to help in the differential diagnosis of Parkinson's Disease and Essential Tremor in early or complex cases. In this paper, we analyze the linear acceleration of the hand tremor recorded with a built-in accelerometer of a smartphone, with a sampling frequency of 100 Hz. This hand tremor signal was thoroughly analyzed to extract different kinematic features in the frequency domain. These features were used to explore different Machine Learning methods to automatically classify and differentiate between control and tremor subjects (HETR Group) and, subsequently, patients with Parkinson's Disease and Essential Tremor (ETPD Group). A sensitiv-

ity of 90.0% and Specificity of 100.0% were obtained with classifiers of the HETR group. On the other hand, classifiers with Sensitivity ranges from 90.0% to 100.0% and Specificity from 80% to 100% were obtained for the ETPD group. These results indicate that the method proposed can be a potential tool to help clinicians with differential diagnoses in complex or early hand tremor cases.

Further information about this work can be found in Appendix B.

Julián D. Loaiza Duque, Antonio J. Sánchez Egea, Theresa Reeb, Hernán A. González Rojas, and Andrés M. González-Vargas. Angular Velocity Analysis Boosted by Machine Learning for Helping in the Differential Diagnosis of Parkinson's Disease and Essential Tremor. In: *IEEE Accesss* 8 (2020), pp. 8886688875. ISBN: 2169-3536. DOI: 10.1109/ACCESS.2020.2993647 [95]

Abstract: Recent research has shown that smartphones/smartwatches have a high potential to help physicians identify and differentiate between different movement disorders. This work aims to develop Machine Learning models to improve the differential diagnosis between patients with Parkinson's Disease and Essential Tremor. For this purpose, we use a smartphone's built-in gyroscope to record the angular velocity signals of two different arm positions during the patient's follow-up, more precisely, in rest and posture positions. To develop and find the best classification models, diverse factors were considered, such as the frequency range, the training and testing divisions, the kinematic features, and the classification method. It was performed a two-stage kinematic analysis, first to differentiate between healthy and trembling subjects and then between patients with Parkinson's Disease and Essential Tremor. The models developed reached an average accuracy of 97.2 ± 3.7% (98.5% Sensitivity, 93.3% Specificity) to differentiate between healthy and trembling subjects and an average accuracy of 77.8 ± 9.9% (75.7% Sensitivity, 80.0% Specificity) to discriminate between Parkinson's Disease and Essential Tremor patients. Therefore, we conclude that the angular velocity signal can be used to develop Machine Learning models for the differential diagnosis of Parkinson's Disease and Essential Tremor.

Further information about this work can be found in Appendix C.

Julián D. Loaiza Duque, Antonio J. Sánchez Egea, Hernán A. González Rojas, Pedro Chána-Cuevas, Joaquim J. Ferreira, and Joao Costa. TremorSoft: A decision support application for differential diagnosis between Parkinsons disease and Essential Tremor. In: *SoftwareX* 22 (May 2023), p. 101393. ISSN: 2352-7110. DOI: 10.1016/j.softx.2023.101393 [96]

Abstract: A cost-effective, non-invasive, and easy-to-use tool is presented that uses the 6-axis inertial sensor of the smartphone or a specific wearable sensor, boosted by machine learning, to support early differential diagnosis of Parkinsons disease and Essential Tremor. A dedicated

web server helps extract the kinematic indexes from the recorded signals, implement the machine learning models and return the resulting classification to the App. Thus, clinicians can use this App as a support tool in the clinic, contributing to performing motor evaluations in the uncertain and undecided stages of the diseases and promoting appropriate, fast, and timely therapeutic responses.

Further information about this work can be found in Appendix F.

### Other scientific collaborations

This subsection includes other external work carried out with collaborators from the department where the research was conducted.

- María Alejandra Cerón, Julián Loaiza Duque, Sergi Barrantes Verdoy, Ana López Ojeda, Marta Alcover Morro, Pedro Quetglas Barea, Hernán A González Rojas, Antonio J Sánchez Egea. Characterization of the volume and thickness of DIEP flap by CTA image processing. In: *2021 22nd Symposium on Image, Signal Processing and Artificial Vision, STSIVA 2021 - Conference Proceedings (2021)*. DOI: 10.1109/STSIVA53688.2021.9592004. [97]
- Erick Chávez Pereda, Julián D. Loaiza Duque, María Alejandra Cerón Hurtado, Hernán A. González Rojas, Antonio J. Sánchez Egea. Erick D.Chávez Pereda et al. Experimental Data-Driven Insertion Force Analyses of Hypodermic Needles in a Soft Tissue with an In-House Test Bench. In: *Communications in Computer and Information Science 1685 CCIS (2022)*, pp. 415422. ISSN: 18650937.DOI: 10.1007/978-3-031-20611-5_34/COVER. [98]

# Chapter 2

# Machine Learning Differentiation of Tremor Disorders in Matlab

In this chapter, the methodology used to perform the acquisition, analysis, and classification of tremor signals in individuals with Parkinson's disease, essential tremor, and healthy subjects is detailed, as shown in Figure 2.1. Figure 2.1 illustrates this approach based on prior research examining hand tremor characteristics [4]. The primary aim of this study was to investigate the possibility of combining previously proposed features and machine learning algorithms for constructing improved classification models designed to categorize tremor signals recorded in the current research. The algorithm was first developed using MATLAB's programming environment (MathWorks Inc., USA). To conduct these tasks, a workstation with an Intel i5-9600K processor running at 3.70 GHz, 16 GB of RAM, and an NVIDIA GeForce GTX 1650 graphics card with 4 GB of video memory (V-RAM).

A thorough data acquisition protocol was implemented to capture tremor signals in resting and postural positions. Signal acquisition utilized a device equipped with both a gyroscope and an accelerometer. Then, preprocessing techniques were employed to guarantee record quality by eliminating potential noise and artifacts. Significant kinematic feature extraction processes were performed on the preprocessed signals to identify and describe tremors. Feature selection was found to be paramount in reducing data complexity and removing non-contributing features for analysis. Non-parametric statistical methods, including the Chi-squared test and the unbiased tree method, were implemented to identify the most discriminative and relevant features. The final phase of this study centered on training and testing classification models. Various kinematic features, training and testing data ratios, and frequency ranges were examined in different combinations. Multiple classification methods were employed to evaluate the performance of each model utilizing metrics

such as sensitivity, specificity, accuracy, and balanced accuracy.

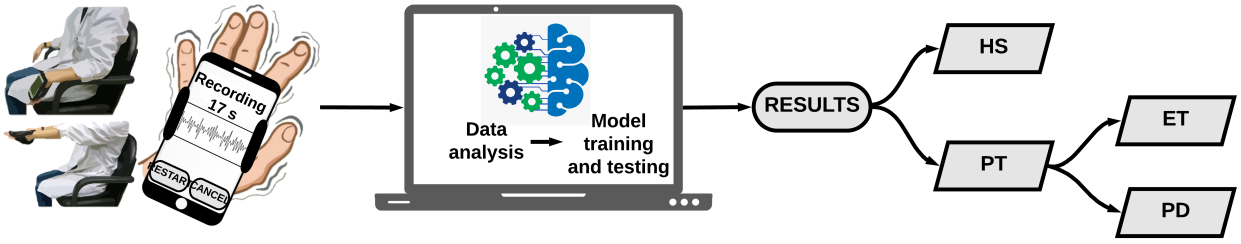



Figure 2.1: Development Outline of MATLAB analysis.

## 2.1 Data Acquisition Protocol and Database Description

A precise data collection protocol was implemented to record tremor signals in individuals with Parkinson's Disease, Essential Tremor, and Healthy Subjects. The data collection process utilized the methodology employed in the research conducted by [4]. During this research, the initial database, which formed the basis of a significant part of this research, was recorded at the *Movement Disorders Unit of the Hospital Clínic Barcelona* between October 2015 and December 2016.

### 2.1.1 Data Acquisition Procedure

Subjects were instructed to sit in a chair with armrests, and a securing strap was used to attach a smartphone or wearable device equipped with a gyroscope and accelerometer to the dorsum of the hand that exhibited the most pronounced tremor signs in patients or to the dominant hand in healthy subjects. Signals from the gyroscope and accelerometer were recorded at a sampling frequency of 100 Hz, and each recording lasted 30 seconds. Two arm positions were measured during the acquisition process: The *Resting position* is when the subject rests their forearms on the chair's armrests, while the *Postural position* involves holding both upper limbs fully extended and parallel to the ground.

### 2.1.2 Informed Consent and Anonymization

All participants provided written informed consent before their inclusion in the study. Codes were assigned to the subjects to maintain confidentiality and anonymity instead of utilizing their actual identities. The *Hospital Clinic Barcelona Ethics Committee* authorized the study in compliance with the ethical principles established in the Declaration of Helsinki and its subsequent amendments.

### 2.1.3 Devices and Apps Used

Tremor signals were recorded utilizing the 6-axis inertial sensor built into an iPhone 5S through the SensorLog recording app. This device and app were chosen due to their capacity to precisely and comprehensively record data at a sampling frequency of 100 Hz.

### 2.1.4 Initial Database

The database initially consists of 51 registered participants, with the following distribution: 19 individuals with Parkinson's Disease, 20 individuals with Essential Tremor, and 12 healthy subjects. All patients in the study displayed visible hand tremors and received definitive diagnoses of either Parkinson's Disease or Essential Tremor or were highly suspected of having these conditions. Patients diagnosed with Essential Tremor scored between 1 and 2 on the Fahn-Tolosa-Marín scale, while those diagnosed with Parkinson's Disease were evaluated using the Unified Parkinson's Disease Rating Scale (UPDRS). In addition, patients with Parkinson's Disease underwent a SPECT test for diagnostic confirmation.

### 2.1.5 Database Expansion

Due to limitations in the size of the subject database registered at the Hospital Clínic Barcelona, efforts were made to expand the database as much as possible throughout this research. To achieve this, collaborations were established with the Chilean NGO, *CETRAM*, and the *ACP* to increase the data. At *CETRAM*, 25 new patients were registered. Among them, 18 had Parkinson's disease, and 7 had Essential Tremor. Additionally, through the Vilanova I la Geltrù delegation of the *ACP*, it was possible to record the data of 22 new patients, including 12 with Parkinson's Disease and 10 healthy subjects, the latter being relatives of patients and members of the association.

The patient registry followed the same methodology as the original database, except for two differences. Firstly, the recordings utilized *Xsens Dot*, a type of wireless inertial sensor that is lighter and more manageable. The smaller size of this device assists in minimizing the potential impact of the recording weight of the device on the data quality. Another benefit of these sensors is the potential to receive data in real-time through Bluetooth once the recording concludes. The recordings were made at a frequency of 120 Hz, which was selected because the device only allows sampling at nine predefined frequencies (1 Hz, 4 Hz, 10 Hz, 12 Hz, 15 Hz, 20 Hz, 30 Hz, 60 Hz, and 120 Hz). The ninth option was selected to record the data because it was the closest value to the one used in the Clinic's database. This ensures equivalent or improved signal resolution within

the same time period as the initial recordings. Integration of the new records with the initial ones considered this variation.

## 2.2 Analysis of the Hand Tremor Data Acquired

Analyzing gyroscope and accelerometer recordings has been crucial for detecting and characterizing tremors associated with Parkinson's disease, Essential Tremor, and Physiological Tremor. According to previous literature [17, 40, 41, 99, 100], as mentioned in section 1.1, it is known that Parkinson's disease manifests with resting tremor in the frequency range of 4 to 6 Hz, while essential tremor presents with postural or kinetic tremor in the range of 5 to 8 Hz, while physiological tremor falls in the range of 8 to 12 Hz. The gyroscope and accelerometer signals were subjected to the following preprocessing, feature extraction, and feature selection procedures to perform the analysis.

### 2.2.1 Signal Preprocessing

Data preprocessing ensures that records are appropriately conditioned for kinematic feature extraction and classification model training processes. This stage encompasses various tasks to clean and condition data. By reducing noise and artifacts, more representative signals of the tremors of interest are obtained, significantly improving the analysis and training of classification models. The following are the main actions performed during signal preprocessing.

- The initial preprocessing step requires removing around 2 seconds from the beginning and end of the signals. This prevents artifacts from being generated at the beginning and at the end of the data recording since, in manual recording, there may be motion records during these signal segments that do not correspond to hand tremors.

- A filtering process is subsequently employed to decrease sensor drift and shifts resulting from physical phenomena like motion artifacts and other interferences [35, 80]. Two 10th-order Butterworth filters with different cutoff frequencies are implemented for this purpose. The initial filter possesses cutoff frequencies ranging from 3 to 10 Hz [16], while the subsequent filter has cutoff frequencies ranging from 1 to 16 Hz [26]. These cutoff frequencies are chosen by considering the characteristic frequencies of the examined tremors. This enables identifying the optimal frequency range for feature extraction, resulting in classification model development with the best possible performance. By implementing this filtering process, the

analysis can concentrate on particular frequencies associated with pathological and physiological tremors, thus enhancing the detection of essential kinematic features.

After the preprocessing stage, the signals were prepared for the subsequent stage, which involved feature extraction. Figure 2.2 illustrates the impact of preprocessing on the angular velocity signals of individuals with Parkinson's Disease and Essential Tremor and Healthy subjects in a postural position before and after signal processing.

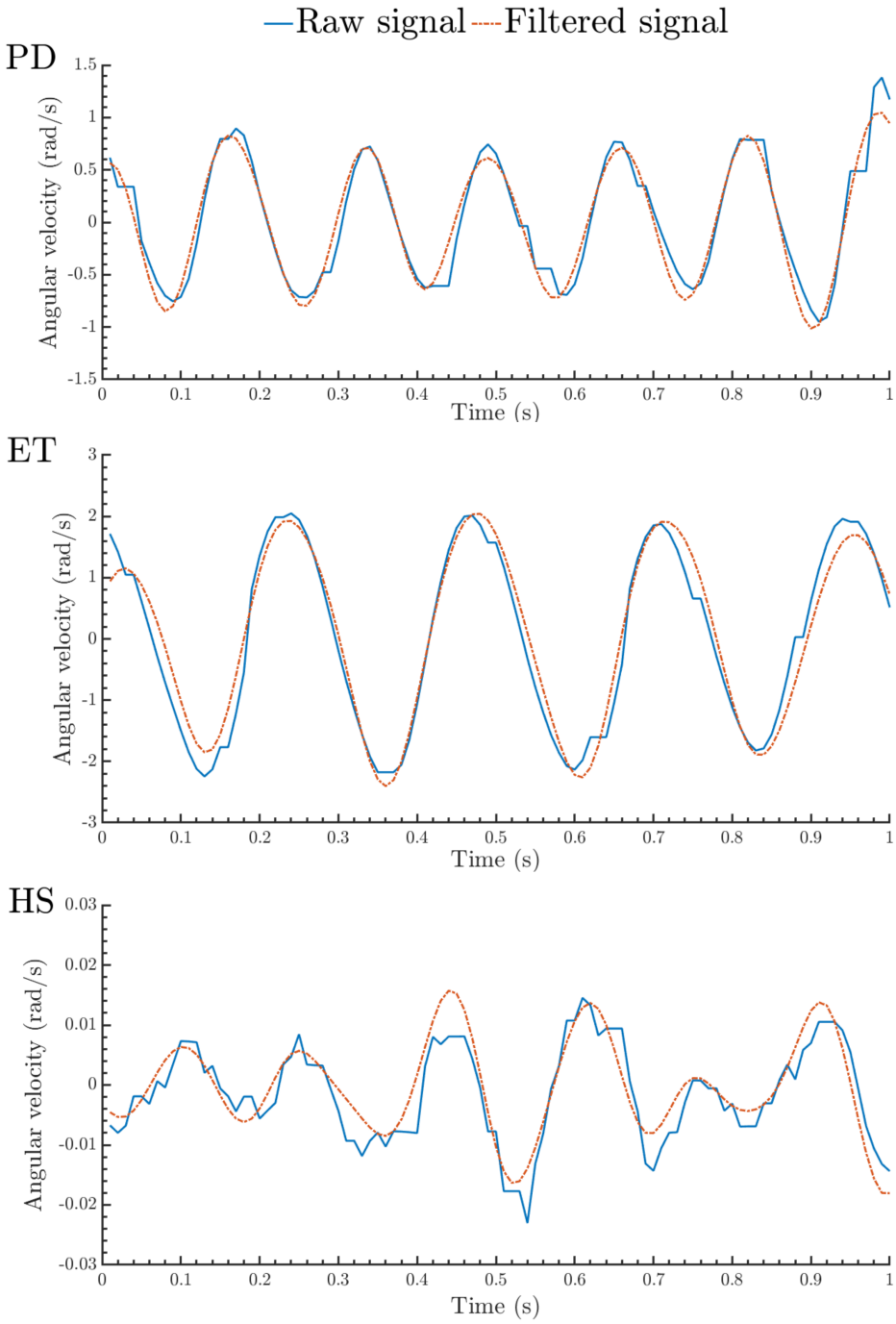



Figure 2.2: Time-domain signal of subjects with Parkinson's Disease and Essential Tremor, as well as healthy subjects in the postural position, before and after signal processing.

### 2.2.2 Kinematic feature extraction

The analysis of features in the frequency domain was accomplished by estimating the Power Spectral Density (PSD) of the preprocessed signals. The PSD was calculated using filtered and trimmed signals from the gyroscope and accelerometer in all three spatial directions (X, Y, and Z). This calculation was performed using the Welch method [101], which involves averaging 3-second signal segments with a Hanning window with a 50% overlap. This produces a power spectral density (PSD) for each spatial direction, which is subsequently averaged to yield a single averaged and scaled PSD. Kinematic features are then computed from the Averaged PSD and used as inputs to the classification models. Figure 2.3 presents the PSD of a subject with Essential Tremor, which has been averaged and scaled, and illustrates the kinematic features computed from the spectral power analysis. Initially, 4 kinematic features are extracted for both arm positions, as detailed in the following description:

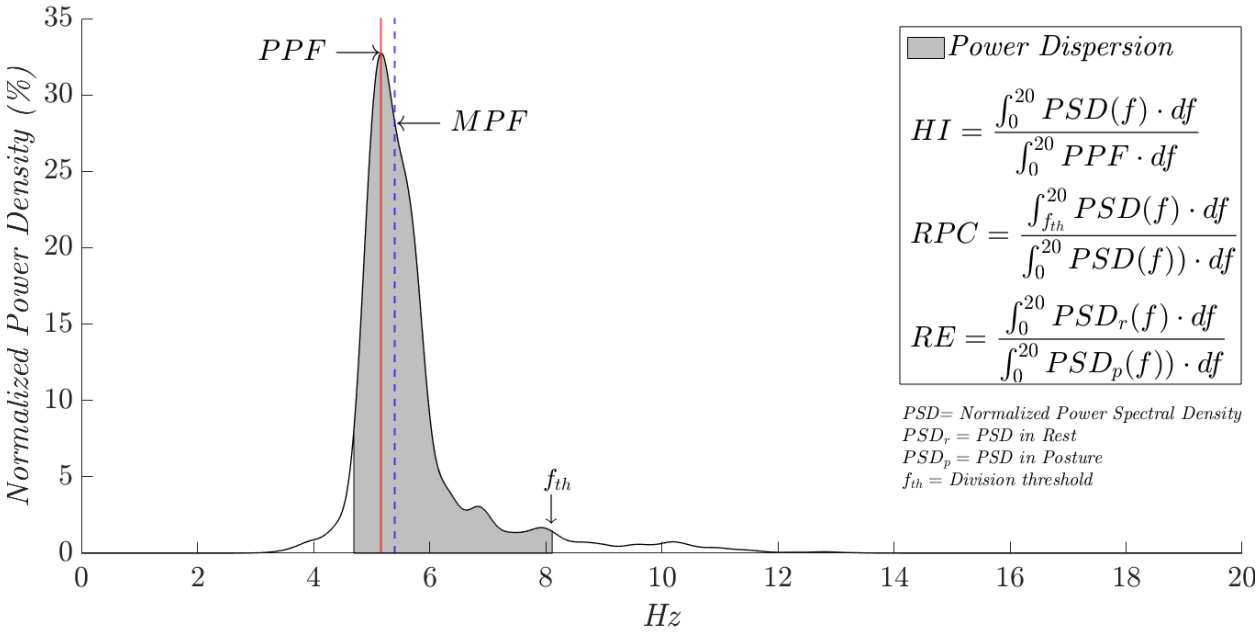



Figure 2.3: Normalized Power Spectral Density of tremor in a subject with Essential Tremor.

Initially, 4 kinematic features are extracted for both arm positions, as detailed in the following description:

- **Median Power Frequency (MPF):** This feature represents the dominant frequency of the tremor oscillations, the frequency at which the PSD decreases to half its maximum value.
- **Power Bandwidth (PB):** Indicates the width of the frequency spectrum of the tremor. The frequency band centered on the MPF that contains 90% of the total signal power.
- **Peak Power Frequency (PPF):** This measure provides information about the dominant

frequency of the most intense oscillations. It represents the frequency of the Peak Power (PP) in the PSD.

- **Harmonic Index (HI):** The HI measures the ratio between the area under the PSD curve and a rectangle bounded by the frequency band of interest ($f_l$ - $f_h$) and the PP. A higher HI indicates a greater energy content in the frequency band of interest.

$$HI = \frac{\int_{f_{th}}^{f_h} PSD(f) \cdot df}{PP \cdot (f_h - f_l)} \tag{2.1}$$

In the study conducted by citebarrantes2017, 2 new kinematic features were proposed: RPC and RE. These features are predicated on the idea that tremor frequency components vary between resting and postural conditions. It is assumed that Parkinson's disease patients should have a higher total spectral power of tremor in the resting position than in the postural position, and vice versa for patients with Essential Tremor. Accordingly, these novel features employ data from both positions in their computation:

- **Relative Power Contribution to the First Harmonic (RPC):** This measurement calculates the correlation between the power of harmonics within a frequency division threshold ($f_{th}$) and $f_h$ and the total power within the area of interest ($f_l$ $f_h$). It offers insight into the existence of harmonics in the signal.

$$RPC = \frac{\int_{f_{th}}^{f_h} PSD(f) \cdot df}{\int_{f_l}^{f_h} PSD(f) \cdot df} \tag{2.2}$$

- **Relative Energy (RE):** This metric assesses the connection between the normalized resting PSD ($PSD_R$) and postural PSD ($PSD_P$) in the frequency range of $f_l$ to $f_h$. It permits the assessment of tremor energy variation between the two arm positions.

$$RE = \frac{\int_{f_l}^{f_h} PSD_R \cdot df}{\int_{f_l}^{f_h} PSD_P \cdot df} \tag{2.3}$$

During this study, two additional features were proposed. These features, like the previous ones, consider information from both positions.

- **Harmonic Index Ratio (HIR):** It depicts the correlation between the harmonic indices of the arms at rest and in posture. This enables a contrast of the harmonic composition of the tremor across both arm positions.

$$HIR = \frac{HI_R}{HI_P} \tag{2.4}$$

- **Sum of Maximum Power (SMP):** The sum of the PP value from the rest and posture positions represents the total power in the most dominant frequencies from both positions.

$$SPM = PM_R + PM_P \tag{2.5}$$

#### Labeling of Kinematic Features

After extracting the matrix of kinematic features, each row (or set of features) is labeled based on the subject's condition from whom it was extracted. There are two specific scenarios or cases that are considered:

1. **Case 1: Tremor patients vs. Healthy subjects**
   - (Tremor patient) - Positive class.
   - (Healthy subject) - Negative class.
2. **Case 2: Parkinson's disease vs. Essential Tremor**
   - (Parkinson's disease) - Positive class.
   - (Essential Tremor) - Negative class.

The labeled features in Case 1 were utilized to distinguish between tremor patients and healthy subjects, while the labeled features in Case 2 were utilized to distinguish between Parkinson's disease patients and Essential Tremor patients.

### 2.2.3 Feature Selection of Kinematic features

Feature selection was an essential step in the data analysis process as it reduces the dimensionality of the feature set and eliminates redundant or irrelevant features, thereby enhancing the efficiency and accuracy of classification models [102, 103]. In this study, two well-known methods in the literature were used for feature selection: the chi-squared test and the unbiased tree method [104, 105].

- **Chi-squared test** The Chi-square test is a statistical method employed to assess independence between two variables. This study utilized it to evaluate the degree of dependence between each feature and the classification variable corresponding to the health condition. The features that obtained higher Chi-square values were deemed more closely linked to the classification variable and were selected as the most relevant.

- **Unbiased Tree test** The Unbiased Tree test is a feature selection approach that relies on decision trees. This approach aimed to construct impartial decision trees for each feature and evaluate their significance in categorizing samples into distinct groups. As a result, the features that significantly influence classification were deemed the most critical.

For the Feature Selection process, the following steps were followed:

1. The feature selection methods mentioned earlier were applied to the feature matrix obtained previously, which contains the kinematic feature values for each subject.
2. The top importance values for each feature selection method were determined. Specifically, the ten most relevant features were identified through the Chi-squared test and the Unbiased Tree method.
3. Common features identified in both tests were selected for further analysis. This led to the identification of a subset of features that were found to be the most discriminative and relevant in the classification of study cases, specifically those involving tremor vs. healthy participants and Parkinson's disease vs. Essential Tremor.

The feature selection process was conducted on two frequency ranges- 1 to 16 Hz and 3 to 10 Hz. Thus, two sets of features were selected for each case study- Case 1 and 2. The training time and accuracy can be improved by reducing the number of features used in the model training. This allows for focusing solely on the most relevant features for class differentiation.

## 2.3 Machine Learning-Based Classification Models

This section outlines the methodology used to construct classification models and assess the discriminative capabilities of selected features from the prior phase. These models are built upon four crucial factors: the frequency range under examination, the proportion of training and testing data, the utilized kinematic features, and the classification algorithm applied.

1. **Frequency Range of Analysis:** As previously stated, kinematic features were extracted at two different frequency ranges, 1-16 Hz and 3-10 Hz, in order to determine the optimal range for achieving maximum model performance in both cases.
2. **Training and Testing Data proportion:** The dataset was divided randomly into three different training and testing data proportions: 30/70, 50/50, and 70/30. These divisions

were applied to each classification case: Case 1 (distinguishing pathological tremors from physiological tremors) and Case 2 (differentiating Parkinson's disease from Essential Tremor). To avoid biases in the models, it was ensured that positive and negative classes were evenly distributed in each training and testing set. Table 2.1 presents the class proportions obtained in the training and testing datasets for both cases across all proportions.

Table 2.1: Training and testing set class ratios.

| | Case 1: temblor vs. healthy | | | | Case 2: Parkinson vs. Essential Tremor | | | |
|---|---|---|---|---|---|---|---|---|
| Division | Training | | Testing | | Training | | Testing | |
| (%) | PC | NC | PC | NC | PC | NC | PC | NC |
| 30 / 70 | 12 | 4 | 27 | 8 | 6 | 6 | 13 | 14 |
| 50 / 50 | 20 | 6 | 19 | 6 | 10 | 10 | 9 | 10 |
| 70 / 30 | 27 | 8 | 12 | 4 | 13 | 14 | 6 | 6 |

PC, Positive class. NC, Negative class.

The decision to test with three divisions instead of one was made to assess the impact of data distribution on model performance.

3. **Selected Kinematic Features:** From the extracted features of the data analysis, every conceivable feature combination ranging from one feature to all was identified. However, to avoid complexity and overfitting, a maximum of five features was set for each model. In some cases, up to 31 feature combinations resulted from this constraint. Each combination was evaluated to determine its individual discriminative ability and its performance in combination with other features using the appropriate classification methods.

4. **Classification Method used:** To train the classification models, the *Classification Learner* application in MATLAB was used. This tool offers a range of supervised machine-learning methodologies, such as Decision Trees, Discriminant Analysis, Support Vector Machines, Logistic Regression, k-nearest Neighbors, Naive Bayes, and Ensemble Classification techniques. The application provides default hyperparameter settings for these methods, enabling exploration of up to 25 hyperparameter configurations to achieve optimal performance for each trained model. As a result, 775 distinct classification models were created for certain cases by considering the various configuration settings and potential feature combinations.

Due to the limited and imbalanced dataset, stratified k-fold cross-validation was used to evaluate the performance of the classification models. This approach guarantees that the class distribution in each split matches the distribution in the overall training dataset [106]. Accuracy and area under

the receiver operating characteristic (ROC) curve (AUC) were estimated for each classification model from the classification probabilities obtained by cross-validation. The performance metrics, such as Sensitivity (Eq. 2.6), Specificity (Eq. 2.7), Precision (Eq. 2.8), and Balanced Precision (Eq. 2.9), were calculated utilizing the test set. The equations used for the calculations are presented below.

- Sensitivity: The ability of the model to correctly identify positive cases (tremor in Case 1 or Parkinson's disease in Case 2).

$$Sensitivity = \frac{TP}{TP + FN} \tag{2.6}$$

- Specificity: The ability of the model to correctly identify negative cases (healthy individuals in Case 1 or Essential Tremor in Case 2).

$$Specificity = \frac{TN}{TN + FP} \tag{2.7}$$

- Accuracy: The proportion of correctly classified subjects in the test set relative to the total number of predictions.

$$Precision = \frac{TP + TN}{TP + FP + FN + TN} \tag{2.8}$$

- Balanced Accuracy: A useful metric for imbalanced classes [107], calculated as the average between sensitivity and specificity.

$$Balanced\ Accuracy = \frac{Sensibilidad + Especificidad}{2} \tag{2.9}$$

To ensure a reliable evaluation of the models, the training and testing process was randomly repeated 100 times for identical feature combinations and classification approaches across each of the three training/test splits. The average performance metrics were computed for each model in each iteration. The top three models were selected for Cases 1 and 2, demonstrating the highest accuracy, sensitivity, and specificity levels across the dataset. Figure 2.4 outlines the procedure for creating and choosing classification models.

## 2.4 Analysis of Linear Acceleration in Tremor Disorders

This section presents the findings from the analysis of Linear Acceleration signals from 51 subjects. To accomplish this, the methodology described in the previous sections was used, where classification models trained on these data showed a sensitivity of 90.0% and a specificity of 100.0%

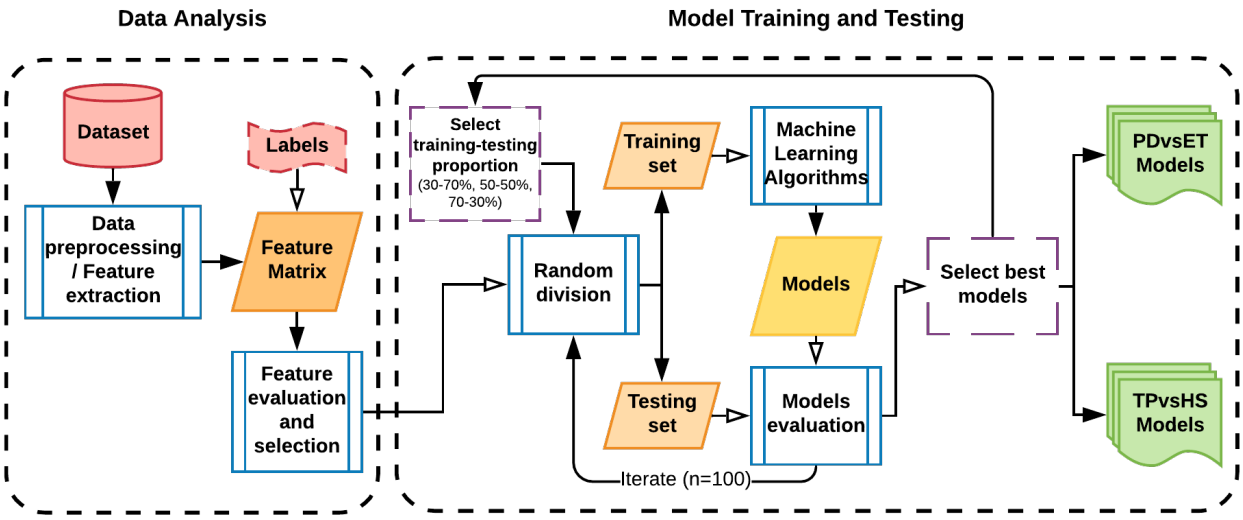



Figure 2.4: Process diagram for the development and selection of classification models.

to discriminate individuals in Case 1, that is, to discriminate tremor subjects from healthy subjects. On the other hand, the classifiers produced sensitivities ranging from 90.0% to 100.0% and specificities from 80.0% to 100.0% for Case 2, which involves differentiating between Parkinson's disease and Essential Tremor. Refer to Appendix I for a more detailed explanation of the specific methodology. Figure 2.4 depicts the tasks carried out during the process of training, validating, and selecting classification models. The data were randomly split into two sets (training set and validation set) in a 70:30 ratio. For the training set, 63 combinations of machine learning classification methods and features were tested, including 23 with Principal Component Analysis (PCA) and 23 without PCA.

The classification models' performance was evaluated using 6-fold cross-validation. Each model's accuracy and area under the receiver operating characteristic curve (AUC-ROC) were determined using non-parametric ROC features and classification probabilities from cross-validation. Subsequently, the performance metrics were computed using the validation sets. In this context, Sensitivity refers to the classification model's capability to detect a positive case, that is, patients with tremors in Case 1 or patients with Parkinson's disease in Case 2. Similarly, Specificity pertains to the classification model's ability to identify negative cases, that is, healthy subjects in Case 1 or patients with Essential Tremor in Case 2.

To ensure reliability, 100 repetitions of validation and training were conducted using the same combinations of features and classification methods. This process ensured that training was conducted using diverse datasets to evaluate classification models with different performance levels. Following all iterations, the classification models with the highest mean values of Balanced Accuracy in Cases 1 and 2 were identified. The top ten classification models were listed and examined

in the results section after the 100 iterations.

### 2.4.1 Results

The outcomes of applying the iterative methodology are presented in Table 2.2. All the reported results were acquired through validation set testing. From a pool of 2898 models based on feature and classification method combinations, the top 10 models are listed. The "Average" column denotes the average behavior over 100 iterations where training and validation data were randomized. The "Best Case" column highlights the best performance among all iterations.

Notably, all models utilized the Quadratic Discriminant technique, and MPF was the common kinetic feature for all models. This indicates that the MPF feature can significantly discriminate tremor patients from healthy subjects. The model for classification with the highest average BAcc value is achieved by utilizing MPF and PPF features. In other words, this particular classification model displayed superior performance compared to other models during most of the 100 iterations. The top of the table displays models with the highest BAcc values. These models exhibit favorable combined results in sensitivity and specificity. Note that the top-performing cases for all classification models demonstrate a BAcc value of 95.0% (90.0% sensitivity and 100.0% specificity). In Case 1, Table 2.2 displays the top ten classification models with the highest BAcc values, highlighting which methods and kinetic features are crucial in discriminating between patients with Parkinson's Disease and Essential Tremor.

The initial six classification models implement the Logistic Regression approach, and the four final ones utilize various KNN algorithms. It is important to note that the kinematic feature "RE" is incorporated into all models to distinguish between the two groups, which aligns with the findings from [4] demonstrating this feature's substantial ability to differentiate between Parkinson's Disease and Essential Tremor patients (accuracy of 84.4%). RPC is also a significant feature in the previous article, and the classification models that use these two kinetic features rank among the top five. In this study, the best classification model, which combined the Logistic Regression method and these two kinetic features (RPC and RE), achieved a BAcc value of 100.0% (100.0% sensitivity and 100.0% specificity). Table 2.2 displays five other situations where a BAcc value of 100.0% was reached. These results show promise for creating a decision-support tool to aid physicians in the differential diagnosis of Parkinson's Disease and Essential Tremor. Validation of these findings will require a larger database.

Table 2.2: Parkinson's Disease vs. Essential Tremor patient discrimination. Top 10 Classification Models with the highest BAcc values in Case 1. PCA: Principal Components Analysis, Sen: Sensitivity and Spe: Specificity

| Features | Method | PCA | Averages | | | Best Case | | |
|---|---|---|---|---|---|---|---|---|
| | | | Sen | Spe | BAcc | Sen | Spe | BAcc |
| RPC+RE+HI | Logistic Regression | No | 69.2 | 85.4 | 77.3 | 100.0 | 90.0 | 95.0 |
| RPC+RE+MPF+HI | Logistic Regression | No | 69.5 | 83.0 | 76.3 | 100.0 | 90.0 | 95.0 |
| RPC+RE+PPF+HI | Logistic Regression | No | 67.7 | 84.5 | 76.1 | 100.0 | 80.0 | 90.0 |
| RPC+RE | Logistic Regression | No | 66.7 | 84.8 | 75.8 | 100.0 | 100.0 | 100.0 |
| RPC+RE+PPF | Logistic Regression | No | 66.4 | 84.9 | 75.7 | 90.0 | 90.0 | 90.0 |
| RE+PPF+HI | Logistic Regression | No | 66.6 | 83.9 | 75.3 | 100.0 | 100.0 | 100.0 |
| RE | Medium KNN | No | 73.8 | 76.4 | 75.1 | 100.0 | 100.0 | 100.0 |
| RE | Cubic KNN | No | 73.8 | 76.4 | 75.1 | 100.0 | 100.0 | 100.0 |
| RE | Medium KNN | Yes | 73,8 | 76.4 | 75.1 | 100.0 | 100.0 | 100.0 |
| RE | Cubic KNN | Yes | 73.8 | 76.4 | 75.1 | 100.0 | 100.0 | 100.0 |

### 2.4.2 Conclusions

This section explored the potential benefits of using machine learning to classify patients with hand tremors. The findings indicate that linear acceleration can supply significant data for accurately distinguishing between healthy subjects and those experiencing tremors, ultimately allowing for the differentiation between individuals with Parkinson's Disease versus Essential Tremor. The effectiveness of this differentiation relies heavily on accurately selecting and evaluating the appropriate classifier to implement. Additionally, exceptional performance can be distinguished by utilizing combinations of kinematic features and classification methods throughout classifier training. Combining the Quadratic Discriminant method with the MPF feature demonstrated the most significance in distinguishing between healthy and pathological subjects. Similarly, the Logistic Regression method, in conjunction with the RE and RPC features, was found to be crucial in differentiating between patients with Parkinson's Disease and Essential Tremor.

The section below examines angular velocity signals recorded using a smartphone's gyroscope to assess the efficacy of models trained with kinetic features from these signals in achieving performance levels commensurate with or superior to those derived from linear acceleration signals.

## 2.5 Analysis of Angular Velocity in Tremor Disorders

In this section, the results obtained in evaluating the influence of angular velocity are described in order to identify the combination of kinematic features and machine learning models that allow an optimal differentiation of the subjects studied. A kinematic analysis was carried out in two stages, similar to the one for linear acceleration: first to discriminate between subjects with tremors and healthy subjects, and then to discriminate between patients with Parkinson's disease and Essential tremor. The models achieved an average accuracy of 97.2±3.7% for distinguishing between healthy subjects and those with tremors, with a sensitivity of 98.5% and a specificity of 93.3%. Moreover, for differentiating between patients with Parkinson's disease and Essential Tremor, they obtained an average accuracy of 77.8±9.9% with a sensitivity of 75.7% and a specificity of 80.0%. Detailed information regarding the methodology is presented below. These results are presented in two separate subsections.

The findings are presented in two separate subsections. The first subsection evaluates the ability of the model to discriminate between individuals with and without tremors. Second, the model's capability to differentiate between patients diagnosed with Parkinson's Disease and Essential Tremor is evaluated. It is important to note that during the time of this analysis and paper publication (see Appendix C), the letters A and B were utilized to reference the data recorded during Rest (R) and Posture (P), respectively.

### 2.5.1 Differentiation of patients with tremors and healthy subjects

Table 2.3 displays the evaluation and selection results of features used to distinguish between tremor and healthy subjects. The top five features from the frequency analysis between 3 to 10 Hz were identical in both tests and included SMP, $RPC_P$, $HI_P$, $HI_R$, and $PB_P$. Similarly, the frequency analysis from 1 to 16 Hz identified four out of the top five features that coincided in both tests: SMP, $RPC_P$, $HI_P$, and $PB_P$.

The top left section of Figure 2.5 exhibits the optimal models for distinguishing tremor and healthy subjects within the 3 to 10 Hz frequency range, arranged by the three training/testing subsets. The highest-performing three models for each split are identified and listed based on their average metrics. The SMP feature is included in all nine models, while $PB_P$, $HI_P$, and $RPC_P$ feature in only two models. The top-performing classification model records an average accuracy of 94.3±5.6% (95.9% sensitivity, 89.5% specificity), while its computational cost is about 6.7±0.7 ms on average. This model achieves its results through a 70/30 split, utilizing the Linear Support

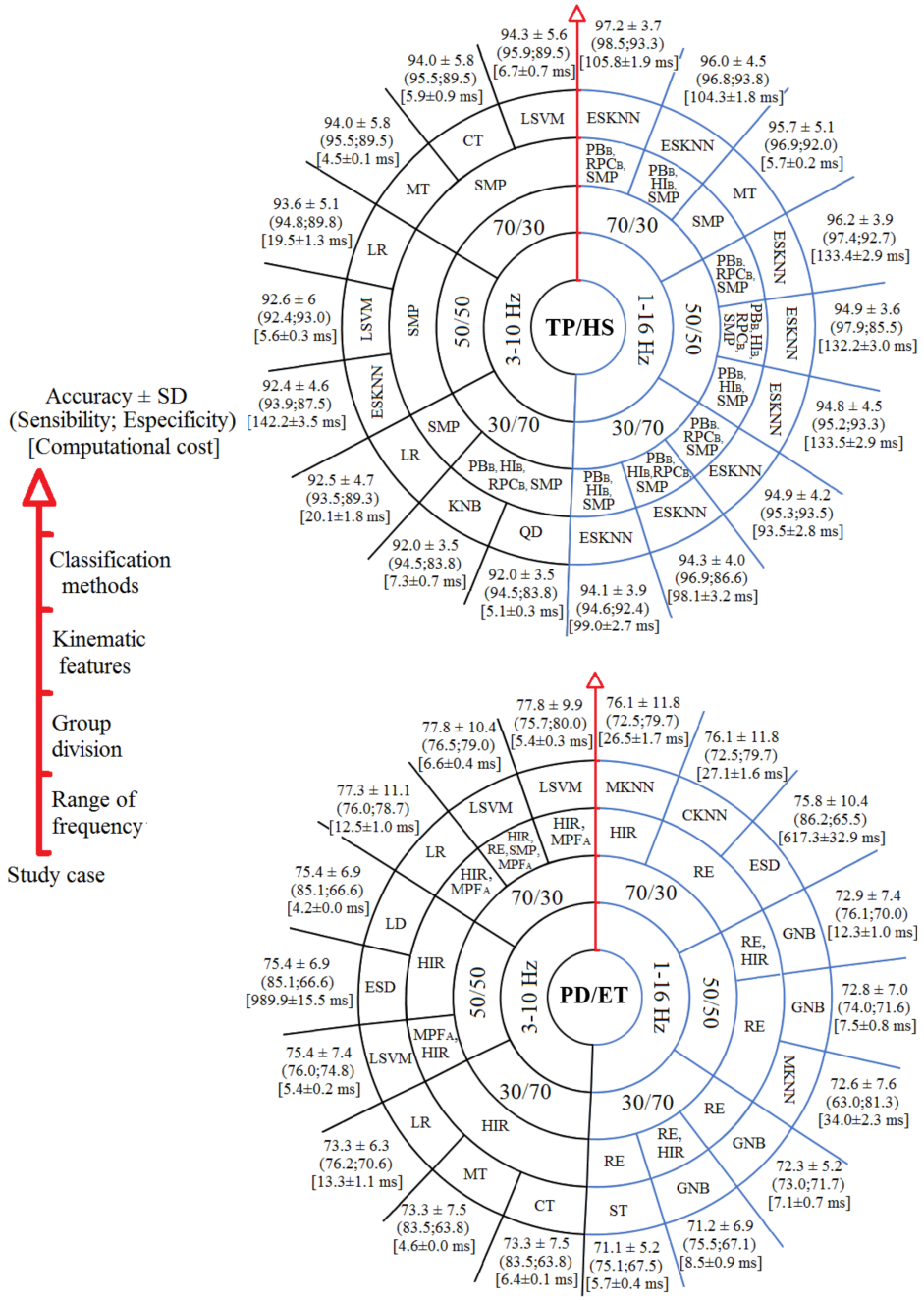



Figure 2.5: Output results of the case study based on machine learning algorithm, frequency range, kinematic features, and classification methods.

ST - Simple Tree, MT - Medium Tree, CT - Complex Tree, LR - Logistic Regression, LD - Linear Discriminant, QD - Quadratic Discriminant, KNB - Naive Bayes with Kernel, GNB - Gaussian Naive Bayes, LSVM - Linear Support Vector Machine, MKNN - Middle Nearest Neighbor, CKNN - Cubic Nearest Neighbor, ESD - Subset Space Discriminant, ESKNN - Ensemble Subset Space Nearest Neighbor.

Table 2.3: **Evaluation and selection of kinematic features for differentiating tremor and healthy subjects.**

| Feature | Position | 3 - 10 Hz | | 1 - 16 Hz | |
|---|---|---|---|---|---|
| | | **CS** | **UT** | **CS** | **UT** |
| MPF | A | 3.23 | 0.04 | 3.02 | 0.07 |
| | B | 3.36 | 0.02 | 3.98 | 0.03 |
| PB | A | 3.98 | 0.03 | 3.98 | 0.06 |
| | B | **8.12** | **0.07** | **11.74** | **0.14** |
| PPF | A | 3.49 | 0.01 | 3.73 | 0.06 |
| | B | 4.76 | 0.03 | 8.12 | 0.08 |
| HI | A | **4.76** | **0.06** | 4.76 | 0.07 |
| | B | **9.01** | **0.08** | **9.01** | **0.14** |
| RPC | A | 3.42 | 0.01 | 4.96 | 0.10 |
| | B | **6.97** | **0.07** | **10.82** | **0.15** |
| RE | A/B | 1.87 | 0.00 | 3.98 | 0.00 |
| HIR | A/B | 0.78 | 0.01 | 0.35 | 0.00 |
| SMP | A+B | **11.74** | **0.12** | **10.82** | **0.16** |

A, Resting position. B, Postural position. CS, Chi-squared test. UT, Unbiased tree method. Bold values correspond to the five features with the most discriminative values in both tests.

Vector Machine method and the SMP feature. Although varied classification methods exist among the nine listed, the best model applies the Logistic Regression method and SMP feature in both 30/70 and 50/50 splits.

On the right side, the figure displays the most optimal models achieved from frequency analysis ranging from 1 to 16 Hz across all training/testing splits. The top three models are then chosen according to their average performance. All models within this frequency range utilize SMP as a discriminatory feature, while eight of them also employ the $PB_P$ feature. The top-performing model yields an average accuracy of 97.2±3.7%, consisting of 98.5% sensitivity and 93.3% specificity. The mean computational cost of the model is 105.8±1.9 ms. A single-feature model, SMP, uses a 70/30 split and the Medium Tree method. The remaining models implement the Subspace Discriminant method and combine multiple features. On average, utilizing the Medium Tree method with a solitary feature results in significantly lower computational costs compared to implementing the Subspace Discriminant method with multiple features in the models.

### 2.5.2 Differentiation of patients with Parkinson's Disease vs. Essential Tremor

Table 2.4 shows the evaluation and selection of features for the differentiation of patients with Parkinson's disease and ET. In the frequency analysis from 3 to 10 Hz, the five features identified in each test separately are the same: SMP, HIR, RE, $RPC_R$, and $MPF_R$. Only three of the five features coincided in the 1 to 16 Hz frequency range: HIR, RE, and $RPC_R$.

The bottom-left part of Figure 2.5 displays the best models for distinguishing Parkinson's Disease and Essential Tremor in the frequency range of 3 to 10 Hz. The top 3 models in each training/test division are listed by their average performance values. The feature HIR appears to provide significant information for differentiating patients with tremors since it is present in all represented models. The overall best performance is achieved in the 70/30 division, combining features HIR and $MPF_R$ and using the SVM method. This model exhibits an average accuracy of 77.8±9.9% (75.7% sensitivity, 80.0% specificity) and an average computational cost of 5.4±0.3 ms.

On the right side of the figure, it visualizes the models with the best performances for distinguishing Parkinson's Disease and Essential Tremor in the frequency range of 1 to 16 Hz. Again, the best model is found in the 70/30 division, with an average accuracy of 76.1±11.8% (72.5% sensitivity, 79.7% specificity) and an average computational cost of 26.5±1.7 ms. The feature presented in most models is RE, which is used in eight of the nine models shown. In the 30/70 and 50/50 divisions, the top two classification models use the Gaussian Naive Bayes method. In contrast, in the 70/30 division, the best performances are obtained with two different configurations of the

Table 2.4: **Evaluation and selection of kinematic features for the differentiation of subjects with tremor: Parkinson vs. Essential Tremor.**

| **Feature** | **Position** | **3 - 10 Hz** | | **1 - 16 Hz** | |
|---|---|---|---|---|---|
| | | **CS** | **UT** | **CS** | **UT** |
| MPF | A | **2,70** | **0,03** | 0,54 | 0,02 |
| | B | 0,13 | 0,00 | 0,12 | 0,00 |
| PB | A | 0,62 | 0,03 | 1,12 | 0,05 |
| | B | 1,12 | 0,02 | 0,62 | 0,01 |
| PPF | A | 0,62 | 0,02 | 0,62 | 0,01 |
| | B | 0,05 | 0,01 | 0,01 | 0,00 |
| HI | A | 0,62 | 0,02 | 0,62 | 0,04 |
| | B | 1,63 | 0,03 | 0,34 | 0,02 |
| RPC | A | **1,91** | **0,04** | **1,37** | **0,05** |
| | B | 0,12 | 0,01 | 1,63 | 0,02 |
| RE | A/B | **3,79** | **0,07** | **5,20** | **0,09** |
| HIR | A/B | **3,34** | **0,07** | **2,10** | **0,06** |
| SMP | A+B | **1,91** | **0,04** | 1,91 | 0,01 |

A, resting position. B, postural position. CS, Chi-square test. UT, unbiased tree method. Values in bold correspond to the five features with the most discriminative values in both tests.

KNN method, achieving the same average accuracy.

Based on the assumption that the frequency components of pathological tremor are higher in either of the two studied positions, the features SMP and HIR were introduced to improve the differentiation between tremor types. In addition, the features RE and RPC, proposed in [4], aim to improve differentiation between patients with Parkinson's and Essential Tremor, as their tremor frequency components vary in resting or postural conditions. Theoretically, individuals with Parkinson's Disease should exhibit higher tremor amplitudes in a resting position as compared to a postural position. Conversely, individuals with Essential Tremor should incur higher tremor amplitudes in a postural position as compared to a rested position. The findings of this study support this, as it appears that the most critical factor in distinguishing between Parkinson's Disease and Essential Tremor patients is the innovative HIR feature. This was implemented in 12 out of the 18 top-performing models as displayed in Figure 2.5. Furthermore, previous works [4, 94] show that the features RE and RPC provide relevant information for distinguishing pathological and healthy subjects across the analyzed frequency ranges. However, our study introduces the SMP feature, which proves to be the most discriminative in several top models. Using this feature alone, high accuracy values were achieved. Upon analysis of the implemented features, it is observed that certain features offer more precise subject differentiation information depending on the case. The models distinguishing subjects in Case 1 primarily comprise features extracted in the postural position. Conversely, Case 2 exhibits a higher concentration of features extracted in the resting position, in accordance with the research of [5, 10].

While the goal is to create high-performing classifiers and prevent classification errors, the machine learning models only use patients with a confirmed diagnosis of Parkinson's Disease and Essential Tremor. Nevertheless, this also implies that the patients are already receiving treatment when they are recorded, so their tremor intensity is notably low. To ensure accurate patient classification, it is crucial to gather additional data from individuals experiencing early-stage tremors to exclude the effects of medication [108] or surgical suppression [109], as these are potential sources of patient misclassification. Furthermore, dataset size presents a significant challenge in developing high-performance models. Since the dataset for training and testing the models is small, the machine learning models implemented in this study are limited in their performance. The dataset needs to be expanded to develop highly accurate models.

## 2.6 Conclusions

The angular velocity signal recorded by the gyroscope and enhanced using machine learning algorithms has proven to be an effective method for differentiating between healthy subjects and patients with tremors, as well as between patients with Parkinson's Disease and Essential Tremor. This differentiation substantially depends on correctly selecting and evaluating classification methods, kinematic features, data processing techniques, and size. The best model for distinguishing tremor and healthy subjects has an average accuracy of 97.2±3.7% (98.5% sensitivity, 93.3% specificity). The average accuracy of the best model for differentiating patients with tremors with Parkinson's Disease and Essential Tremor was 77.8±9.9% (75.7% sensitivity, 80.0% specificity).

During the training of the models, it is able to identify outstanding performance for some combinations of kinematic features, such as SMP, $PB_P$, and $RPC_P$, for differentiating tremor and healthy subjects, as well as HIR and $MPF_R$ for differentiating Parkinson's Disease and Essential Tremor. Regarding classification methods, for differentiating tremor and healthy subjects (Case 1), the best performances are achieved with the SVM and Subset Discriminant method. For the differentiation of patients with Parkinson's Disease and Essential Tremor (Case 2), in the 3 to 10 Hz frequency analysis, the best performance is also obtained with the Linear SVM method. In contrast, in the 1 to 16 Hz range, the best performance is achieved with the Medium KNN method. In both cases, the Linear SVM models have a lower computational cost than the KNN methods.

However, it's possible that in cases where high-performance models were identified in both analyses, they may be overfitting due to the limited amount of data available for model training. The next chapter discusses, among other things, the transition made to implement the algorithms developed in MATLAB in a new programming language. Furthermore, a data augmentation method is implemented with the goal of increasing the amount of data and generating more robust models.

# Chapter 3

# TremorSoft as a decision-support tool for movement disorder assessment

In the previous chapters, an exhaustive analysis of the linear acceleration and angular velocity signals recorded corresponding to the study population's three types of hand tremors was carried out. This analysis aimed to determine whether the information contained in these signals could prove to be significant enough to allow the development and implementation of machine learning models capable of generating a more accurate classification than that obtained by the statistical methods previously used in [4]. The information extracted from these signals was based on specific kinematic features computed in the frequency domain, some of which had been proposed in previous research. Others were defined in the course of this research. Individual analyses of these kinematic features extracted from each type of sensor signal led to the conclusion that it was possible to train classification models capable of discriminating these tremor types with high accuracy using machine learning algorithms. Based on these results and the requirements previously defined in this research for the development process of the decision-support tool, this chapter presents the methodology that was carried out to create an innovative e-health application that promises to make a significant contribution to the field of movement disorders research. This application has been registered under the *Safe Creative License* with the identifier 2206021281741 (see Appendix G) and is called ***TremorSoft***.

Smartphones turned out to be the ideal platform to meet several of the requirements established for developing this tool: simplicity, speed, capacity for real-time implementation during clinical routines, and, last but not least, network connectivity. Therefore, it was decided to develop a mobile application that uses the internal inertial sensors of the smartphone or wirelessly connects to a portable inertial sensor. This will allow the collection of hand tremor data following the

acquisition protocol described in 2.1. Furthermore, given the desire to expand the database with new tremor records from patients with confirmed diagnoses, the application should also allow storing these records in an online database. Therefore, in addition to the mobile application, it was decided to develop a web server or back-end linked to a database to upload and store these new data. The possibility of implementing a web server also allows the tremor calculation and classification tasks to be performed securely, efficiently, and in real time. This, in turn, implies that the classification models can be deployed on the web server.

Up to this point, the MATLAB development environment was very useful for performing the data analysis, training, and testing the classification models. At this stage, it was possible to evaluate the importance or predictive power of the kinematic features defined in subsection 2.2.2 through performance metrics calculated on the developed models (see section 2.3). However, this option does not allow direct implementation of the developed algorithms and models on a smartphone or web server. Therefore, looking for an alternative at the same level or higher than MATLAB was necessary, and the programming language *Python* was chosen. This decision was based on the premise that Python, like MATLAB, is a high-level scientific programming language that enables the development of web applications, software, data science, databases, and more. Python is the ideal alternative for implementing algorithms, classification models, web servers, and databases, in addition to being easy to implement and compatible with multiple platforms. An extensive search was conducted to determine the most appropriate programming language for developing the mobile application. After researching and exploring several options (Java, Kotlin, Dart, Flutter, Swift, among others), the programming language *Kotlin* was chosen to develop the application on Android devices. Kotlin is a simple, cross-platform programming language recommended by *Google* for developing Android applications. Thus, *TremorSoft* was designed as a modular system consisting of a mobile application and a web server connected to a database.

The following section describes the implementation and optimization of data analysis algorithms (preprocessing and feature extraction) in Python.

## 3.1 Python Code: Data Analysis

Appendix D presents the detailed *Python* code developed to carry out the tasks of preprocessing and extraction of kinematic features from the records of 96 subjects in the available databases: 51 subjects registered at the Hospital Clínic Barcelona and 25 patients registered at CETRAM (Chile). The coding process involved searching for and using specific Python libraries, methods, and functions equivalent to those used in MATLAB. To include the CETRAM database in the

data analysis process, functions were introduced in the Python algorithm to select different folders containing the tremor records of the subjects. Each time a folder is selected, the sampling frequency used in the chosen database must be specified. In contrast to the Hospital Clínic database, the new data were recorded at a sampling rate of 120 Hz for approximately 30 seconds in both resting and postural positions. The following are the main changes and incorporations of methods made in the Data Analysis algorithm:

### 3.1.1 Data Transformation of Xsens Dot Sensor Recordings

Since the data sources are different, it was necessary to ensure the equivalence of the data when incorporating the new database, in particular, to ensure that the data were expressed in the same measurement system. In the case of the Clínic database, linear acceleration records are expressed in units of g-force (G), while angular velocity is expressed in radians per second ($rad/s$). On the other hand, the data recorded with the Xsens Dot sensor at CETRAM were recorded in units of square meters per second ($m^2/s$) for linear acceleration and degrees per second ($deg/s$) for angular velocity. The following function performs the unit conversion when the data comes from the Xsens Dot sensors so that it is in the same measurement system as the Clínic database:

Listing 3.1: Data Transformation Function in Python

```
def transform_data(data):
    """
    Transform the data based on the device.

    Args:
        data (pd.DataFrame): Input data DataFrame containing accelerometer and
            gyroscope columns.

    Returns:
        pd.DataFrame, pd.DataFrame: Transformed accelerometer data and gyroscope
            data DataFrames.
    """
    accelerometer_data = data[['Acc_X', 'Acc_Y', 'Acc_Z']]
    gyroscope_data = data[['Gyr_X', 'Gyr_Y', 'Gyr_Z']]

    accelerometer_data = accelerometer_data.copy()
    accelerometer_data['Acc_X'] = accelerometer_data['Acc_X'] * 0.1019716 - 1
    accelerometer_data['Acc_Y'] = accelerometer_data['Acc_Y'] * 0.1019716 - 1
    accelerometer_data['Acc_Z'] = accelerometer_data['Acc_Z'] * 0.1019716 - 1

    gyroscope_data = gyroscope_data.copy()
```

```
gyroscope_data['Gyr_X'] = gyroscope_data['Gyr_X'] * 0.0174533
gyroscope_data['Gyr_Y'] = gyroscope_data['Gyr_Y'] * 0.0174533
gyroscope_data['Gyr_Z'] = gyroscope_data['Gyr_Z'] * 0.0174533

return accelerometer_data, gyroscope_data
```

### 3.1.2 Data Preprocessing Modifications

One of the changes made from the original algorithm was that instead of trimming a 2-second signal segment at the beginning and end of the recorded data, a 20-second segment centered in the middle of the analyzed data was extracted. This new algorithm discarded, on average, about 5 seconds of signal at each end of all recordings (or about 10 seconds of signal in total). This change was made primarily to eliminate possible noise signals caused by human interaction at the beginning or end of the data recording and to analyze all data in the same period. One of the main functions used in this data analysis process is the one that estimates the power spectral density using the Welch method. This function was implemented using the *SciPy* library, a free and open-source Python library used for scientific and technical computing. This library also includes the functions used for data filtering in this process, particularly the functions butter() and *filtfilt()*; the former is used to design the bandpass filter from 3 to 12 Hz, and the latter applies this filter to the studied signals. In addition, to efficiently compute the kinematic features, some functions from two other important and widely known Python libraries, *NumPy* (a Python library specialized in numerical computations and data analysis, especially for large data sets) and *Pandas* (a Python library specialized in manipulating and analyzing data structures) were imported and implemented.

### 3.1.3 Power Band (BP) feature

It is important to note that a new feature was introduced at this stage. However, it was already implicitly calculated: the **Power Band (BP)**, defined as the area under the curve of the power spectral density within the frequency range of interest ($f_l$ - $f_h$). Although $BP$ had previously been used to compute $RE$ and other features, it had not been considered discriminative in differentiating these two pathologies.

$$BP = \int_{f_l}^{f_h} PSD_X \cdot df \tag{3.1}$$

where $X$ would indicate the evaluated position, which can be $R$ *(Rest)* or $P$ *(Posture)*.

### 3.1.4 Data Augmentation and Feature Extraction

Considering that, despite having new data and having registered a total of 76 subjects, it still represents a limited amount of data to develop high-performance classification models without incurring overfitting or underfitting. For this reason, in this stage, a data augmentation method has been incorporated using the *Overlapping Sliding Windows* technique. This technique, widely used in signal processing and sequential data, allows for dividing a continuous sequence of data into multiple smaller segments or windows that partially overlap with each other. Its application is common when analyzing or processing sequential data, as in the current context.

The main idea behind Overlapping Sliding Window*s* is to divide a long sequence into smaller segments for analysis, ensuring that these segments partially overlap and do not lose important information at transitions between them. This technique becomes particularly valuable when working with data exhibiting temporal features or patterns that do not align perfectly with window boundaries. This enriches the analysis by considering not only the static aspects of the signals but also their dynamics and changes over time. The application of this technique as a Data Segmentation method can provide various benefits:

- **Data Augmentation:** More samples are generated from the original data by subdividing long records into shorter segments. Each $X$-second segment is considered a new data instance, increasing the total data available to train the models.
- **Improved Generalization:** Models trained on shorter segments can learn short-term patterns precisely rather than relying solely on long-term patterns. This can enhance the model's generalization ability, making it more robust under various conditions.
- **Capturing Temporal Changes:** As mentioned earlier, when working with shorter segments, it is possible to capture faster temporal changes or fluctuations in the data that might go unnoticed in longer records.

The main steps to implement this *Overlapping Sliding Windows* technique in *Python* are described below:

1. **Parameter Definition:** The window size (*window_size*) was calculated to divide the records of each subject into 5-second segments. This value was determined based on the sampling frequency at which the data were captured (100 or 120 Hz), which determines how many data points will be included in each window. Additionally, a 25% overlap value was defined to determine how many data points will overlap between consecutive windows. Using this

overlap ensures proper capture of temporal changes and results in 532 segments generated from the 76 subjects in both databases.

Listing 3.2: Parameters of the Overlapping Sliding Window method

```
# Sliding window method parameters
window_duration = 5
# seconds
overlap = 0.50
# 50% overlap

# Calculate window size and step size
window_size = int(window_duration * fs)
step_size = int(window_size * (1 - overlap))
```

2. **Window Creation:** A while loop was implemented to traverse the data sequence, extracting data segments of the current window size and processing them. Then, the window is moved forward by a number of data points equal to the overlap value, and the next segment is taken. This process repeats until the entire sequence is covered.

Listing 3.3: Data augmentation using the Sliding Window method

```
# Apply data augmentation using the sliding window method
segments = []
start = 0
while start + window_size <= len(filtered_data):
    segment = filtered_data[start:start + window_size]
    segments.append(segment)
    start += step_size
```

3. **Processing Each Window:** Next, a for loop takes each segment to estimate the *Average Power Spectral Density (PSD)* of the linear acceleration and angular velocity signals, as well as the *PSD* of the components or axes of both signals (2 Signal Types x 3 Axes): Linear Acceleration (aX, aY, aZ) and Angular Velocity (vX, vY, vZ). Each *PSD* provides 17 features, with 7 features for each position (PB, MPF, PBW, PPF, PP, HI, and RPC) and 3 common features (RE, HIR, and SMP). This process multiplied the number of calculated features, resulting in 136 features to train the new models. This change allows the evaluation of the contribution of the signal components individually and collectively.

Listing 3.4: Looped PSD Calculation and Feature Extraction

```
for i, segment in enumerate(segments):
```

```
        frequencies, psd_segment = welch(np.transpose(segment), fs=fs, window
            ='hamming', nperseg=100, nfft=30000, scaling='density', detrend=
            False)

        psd_segment = np.transpose(psd_segment)

        dif_low = np.abs(frequencies - low_cut)
        dif_high = np.abs(frequencies - high_cut)
        low_idx = np.where(dif_low == dif_low.min())[0][0]
        high_idx = np.where(dif_high == dif_high.min())[0][0]

        for axis, psd_values in zip(['', 'x_', 'y_', 'z_'],

    [psd_segment.mean(axis=1)] + [psd_segment[:, i] for i in range(3)]):
            # Code continues here...
```

4. **Concatenation of Results:** The results obtained from each segment are concatenated and labeled into a larger data structure, representing the processed features or segments from the entire original sequence. This is repeated for all records in both databases using the `process_each_file()` function, concatenating all features from the 380 segments.

In summary, this technique has proven to be especially useful for creating a larger and more diverse dataset, generating multiple examples from the original data sequences, each focused on a different part of each sequence. This can enhance the ability of machine learning models to capture relevant patterns and features in the data.

## 3.2 Python Code: Model Training and Testing

This section describes the methodology for performing model training, testing, and evaluation using Python and various libraries. The primary goal of the experiment was to construct and fine-tune predictive models for classifying hand tremor data samples in the two cases evaluated. This section first describes the libraries and modules imported for this purpose. The process is then divided into several key sections, each addressing specific tasks such as data preparation, model fitting and ensemble, model blending, and final model evaluation. The Python code provided in Appendix E exemplifies the overall workflow.

### 3.2.1 Importing Necessary Libraries and Modules

The first step in the experimental process was to import essential Python libraries and modules. These libraries facilitated various operations such as data handling, model building, and evaluation. The critical libraries used in this experiment were

- `pandas` (abbreviated as `pd`): This library was used extensively for data manipulation and analysis. It provided fundamental data structures, such as data frames, which are crucial for managing tabular data.
- `scikit-learn`: A Python machine-learning library that provides data preparation, model selection, and evaluation tools. In this experiment, the `train_test_split` function was used to split the dataset into training and test sets.
- `PyCaret`: A Python library, similar to the MATLAB *Classification Learner App*, designed to automate the end-to-end machine learning process. PyCaret simplifies several tasks: data preparation, model selection, and hyperparameter tuning.

### 3.2.2 Preparing the Experiment Parameters

Before starting the experiment, it was essential to define its parameters. These parameters controlled various aspects of the experiment, including data handling, feature selection, and model training. The `prepare_params()` function was responsible for this task. Key parameters considered in this experiment include:

- `data`: The data set used for training and evaluation.
- `target`: The target variable to predict, labeled as `label` in the data set.
- `normalize`: An indicator indicating whether to perform feature normalization.
- `feature_selection`: A flag indicating whether feature selection techniques should be applied.
- `remove_outliers`: A flag indicating whether outliers should be removed.
- `train_size`: The percentage of the data set used for training (80% in this case).
- `data_split_stratify`: An indicator of whether the data split should be stratified.
- `fold_strategy`: The strategy used for cross-validation, in this case `stratifiedkfold`.

- `fold`: The number of folds used in the cross-validation (in this case, 5).
- `n_jobs`: The number of CPU cores used (-1 indicates that all available cores are used).

The PyCaret experiment setup was configured using the prepared parameters. This setup defined the overall configuration for data preparation, model selection, and evaluation.

### 3.2.3 Model Training, Fine-tuning, and Ensemble

Optimization parameters were set before the experimental process began, specifying the metric to be optimized, 'F1', and the number of iterations for hyperparameter optimization (50 iterations). Once these parameters were set, the experimental process consisted of constructing predictive models, fine-tuning their hyperparameters, and combining them to improve predictive performance.

$$F1 = \frac{2 \cdot (Precision \cdot Recall)}{Precision + Recall} \tag{3.2}$$

All available classification models were constructed, and the top five models were selected based on the optimization metric ('F1') using the `compare_models` function of PyCaret. These models were fine-tuned, combined, and mixed, resulting in several model ensembles.
The `tune_and_ensemble_model()` function encapsulated these tasks. The following steps were performed:

1. **Bagging:** The tuned model has been further improved by ensemble combination, which creates multiple instances of the model with different subsets of training data.
2. **Boosting:** Boosting was applied to the fine-tuned model, creating an ensemble of models that sequentially correct the errors of their predecessors.
3. **Blending:** Model blending is a technique that combines multiple models to create a single predictive model. The `blend_models_sets()` function was used for this purpose. It blended models based on a given method index corresponding to different model combinations. The blending was done using the PyCaret function `blend_models`. The optimized metric 'F1' was used for blending, ensuring that the blended model maximized the F1 score.

### 3.2.4 Model Evaluation and Selection

After building, tuning, and blending the models, the best-performing model was identified using the `automl` function of PyCaret, which selects the model with the highest F1 score. The identified

best model was then calibrated to improve its performance further. The process of training and testing the models concluded with the evaluation and finalization of both the best model and the calibrated best model. These models generated predictions for unseen data, completing the experimental process. The `evaluate_and_finalize_models()` function performed these tasks. Key steps included:

1. **Evaluation:** The model's performance was evaluated using `evaluate_model` function of PyCaret, providing information on various metrics and visualizations.
2. **Finalization:** The best model was finalized using the `finalize_model` function of PyCaret, preparing the model for prediction.
3. **Predictions:** The finalized model was used to predict the test and unseen data.

This model training and testing process concluded with a comprehensive evaluation of the models and selection of the most appropriate final model for the classification task.

## 3.3 Machine Learning Performance

This section presents the results obtained after executing the two previously described processes. As mentioned in the previous section, data augmentation was performed using the Sliding Window with Overlapping technique during the Data Analysis process. A window of 5 seconds with a 25% overlap was configured, generating a new database with 380 subsamples or segments, each properly labeled according to its origin. For example, if a group of segments originated from hand tremors of a patient with Parkinson's disease, each of these segments was labeled as 'PD'. This process was applied to all subsegments of the original signals. The resulting new database was saved in a CSV text file for subsequent use. Below is a detailed description of the methodology used to train classification models using the PyCaret library in this research context. The main objective of this methodology was to develop robust and effective classification models to address the classification problem in the dataset under study.

### 3.3.1 Data Preparation and Configuration

The process began with loading the dataset from the CSV file named "features.csv." Subsequently, a critical preprocessing step was performed in which rows of the dataset that did not align with the target label of interest, in this case, "HS," were filtered out, and the indices were reset to

maintain data integrity. Features (`X`) and the target variable (`y`) were then separated for further analysis. Once the data was prepared, it was divided into training sets (`X_train, y_train`) and test sets (`X_test, y_test`). This division allocated 85% of the data for training and 15% for testing to ensure representativeness in both partitions. Stratification was applied to preserve the class distribution in both partitions and mitigate potential biases. Two fundamental DataFrames were created from these sets: the training data DataFrame (`training_data`) and the unseen data DataFrame (`unseen_data`). In both cases, the corresponding label was added to each observation to facilitate analysis and the subsequent evaluation of models.

To ensure an efficient and effective training process, experiment parameters were configured in PyCaret. These parameters included the definition of training data, specification of the target variable, feature normalization, feature selection, cross-validation strategy (using Stratified K-Fold with 5 folds), and utilizing all available cores to accelerate processing. The choice of the "F1" metric as the primary performance indicator was critical in training classification models. Furthermore, the number of iterations for hyperparameter tuning was set to 50 to explore the hyperparameter space of the models thoroughly.

### 3.3.2 Training, Optimization, and Model Selection

Once the aforementioned steps were completed, training was initiated for various available classification models. The "F1" metric was used to compare and select the top five models that exhibited the best performance. Table 3.1 displays the performance metrics obtained after training each of the available models using the kinematic features of the new dataset. From this table, the top five models were selected in descending order: Extra Trees Classifier (ETC), Light Gradient Boosting Machine (LightGBM), Random Forest Classifier (RFC), Extreme Gradient Boosting (XGBoost), and Gradient Boosting Classifier (GBC). A series of additional steps were undertaken for each of these top five models. Firstly, the model was fine-tuned through an exhaustive hyperparameter search that included 50 iterations. Subsequently, ensemble models were created using Bagging and Boosting techniques based on the tuned model to enhance performance and robustness further.

To determine the optimal model, the "automl" module was applied, which selected the best model based on the "F1" optimization metric. The best model identified underwent comprehensive evaluation using various performance metrics. Finally, the model was finalized, prepared for future deployment, and used to make predictions on the unseen data (unseen_data). Figure 3.1 represents the ROC curve of the optimal model, which achieved a value of 0.99 for both classes.

Table 3.2 summarizes the performance metrics estimated when using the optimal model, the

Table 3.1: Model Performance Metrics with Data Augmentation: 5 seconds window and 25% of overlapping

| Model | Algorithm | Accuracy | AUC | Recall | Precision | F1 | Kappa | MCC |
|---|---|---|---|---|---|---|---|---|
| et | Extra Trees Classifier | 0.9677 | 0.9820 | 0.9600 | 0.9846 | 0.9719 | 0.9337 | 0.9348 |
| lightgbm | Light Gradient Boosting Machine | 0.9401 | 0.9713 | 0.9600 | 0.9392 | 0.9488 | 0.8766 | 0.8787 |
| rf | Random Forest Classifier | 0.9261 | 0.9670 | 0.9360 | 0.9372 | 0.9356 | 0.8490 | 0.8516 |
| xgboost | Extreme Gradient Boosting | 0.9262 | 0.9662 | 0.9360 | 0.9365 | 0.9356 | 0.8491 | 0.8506 |
| gbc | Gradient Boosting Classifier | 0.9121 | 0.9664 | 0.9120 | 0.9356 | 0.9225 | 0.8208 | 0.8236 |
| ada | Ada Boost Classifier | 0.9030 | 0.9455 | 0.9040 | 0.9280 | 0.9132 | 0.8035 | 0.8091 |
| dt | Decision Tree Classifier | 0.8889 | 0.8887 | 0.8880 | 0.9202 | 0.9024 | 0.7733 | 0.7769 |
| qda | Quadratic Discriminant Analysis | 0.8569 | 0.9035 | 0.8640 | 0.8865 | 0.8749 | 0.7075 | 0.7083 |
| lr | Logistic Regression | 0.7600 | 0.8556 | 0.8240 | 0.7795 | 0.7960 | 0.5019 | 0.5119 |
| knn | K Neighbors Classifier | 0.7557 | 0.8458 | 0.7760 | 0.7909 | 0.7805 | 0.5057 | 0.5120 |
| svm | SVM - Linear Kernel | 0.7416 | 0.0000 | 0.7600 | 0.7885 | 0.7715 | 0.4729 | 0.4772 |
| ridge | Ridge Classifier | 0.7140 | 0.0000 | 0.8080 | 0.7266 | 0.7637 | 0.4022 | 0.4097 |
| lda | Linear Discriminant Analysis | 0.7001 | 0.7721 | 0.7840 | 0.7219 | 0.7501 | 0.3753 | 0.3805 |
| dummy | Dummy Classifier | 0.5761 | 0.5000 | 1.0000 | 0.5761 | 0.7310 | 0.0000 | 0.0000 |
| nb | Naive Bayes | 0.6037 | 0.7596 | 0.5920 | 0.7184 | 0.6053 | 0.2000 | 0.2381 |

Extra Trees Classifier, to predict Test Data and Unseen Data. The model can be observed to classify with 96.36% accuracy (53 out of 56 samples) on the Test Data and 93.75% accuracy (45 out of 48 samples) on the Unseen Data, demonstrating high performance and low bias.

Table 3.2: Performance of the Best Model on Test and Unseen Data

| | **Extra Trees Classifier / 5-second window / 25% Overlapping** | | | | | | |
|---|---|---|---|---|---|---|---|
| **Data** | **Accuracy** | **AUC** | **Recall** | **Precision** | **F1** | **Kappa** | **MCC** |
| Test | 0.9636 | 0.9864 | 0.9636 | 0.9665 | 0.9638 | 0.9262 | 0.9287 |
| Unseen | 0.9375 | 0.9893 | 0.9375 | 0.9378 | 0.9372 | 0.8705 | 0.8713 |

It is important to note that feature selection was enabled in the PyCaret experiment configuration without specifying a fixed number of features. In this case, the process selected the 28 most important features. Figure 3.2 displays the features selected during the experiment. The influence of the information recorded in the resting position is noteworthy, as the top 10 most significant features are derived from this position. The newly incorporated feature, Band Power (BP), also demonstrates significant potential for developing more robust models. The influence and potential of the Harmonic Index (HI) feature are also evident, as it ranks among the top 10 features.

Furthermore, it is observed that the model presents a balanced representation of features ex-

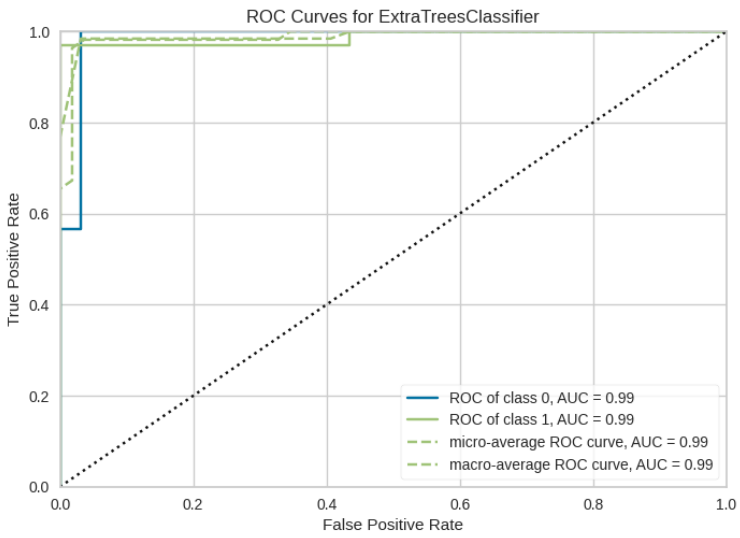



Figure 3.1: Model ROC Curve

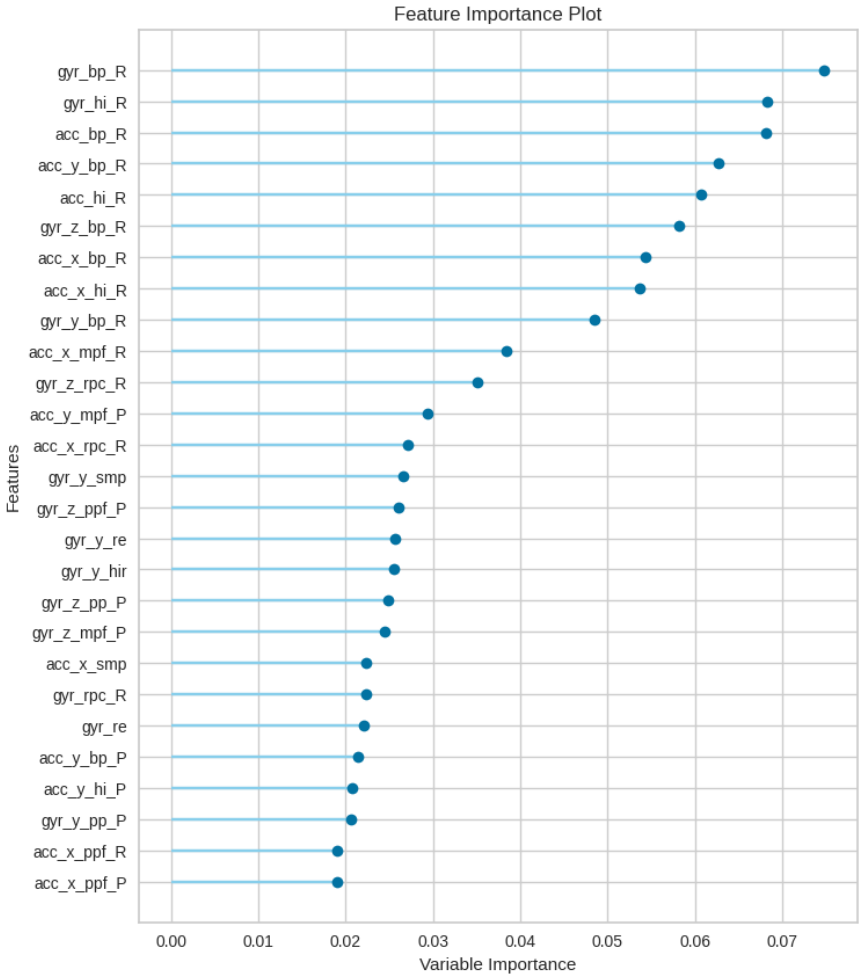



Figure 3.2: Model Feature Importance

tracted from each type of sensor, as well as a proportional distribution of features extracted individually in each axis and features derived from the Average PSD.

## 3.4 Software Architecture

TremorSoft is a tool for decision support that utilizes the 6-axis inertial sensor of a smartphone or a wearable sensor dedicated and connected via Bluetooth for recording, analyzing, and categorizing hand tremor data. The tool comprises two components: firstly, a smartphone app that gathers demographic, clinical, and kinematic data of the subject under evaluation, and secondly, a web server that further processes the collected kinematic data via machine learning models to differentiate between control, essential tremor, and Parkinson's disease, based on the features extracted from the data. The application was developed on the Android operating system, utilizing Kotlin and the *Android SDK* in conjunction with the *Xsens DOT SDK* to serve as the front-end tool. The target version for the operating system is Android 8.0 Oreo or above. The back-end web server runs processes on the Heroku platform, programming in Python. The `Retrofit network` library acts as a REST client for uploading and retrieving data from the back-end. The Firebase platform is used for authentication services. Refer to Figure 3.3 for an overview of the software architecture.

### 3.4.1 Software funcionalities

#### Front-end

The Android application is a front end, enabling users, doctors, and movement disorder specialists to document pertinent information on patients with suspected or diagnosed Parkinson's or Essential Tremor. The scope of recorded data covers primary clinical data, hand tremor signals, and diagnosis and treatment information for diagnosed patients. The mobile application instructs users to record hand tremor signals using either the smartphone's built-in inertial sensors (gyroscope and accelerometer) or an external inertial sensor (Xsens DOT) connected wirelessly. The recorded signals correspond to two positions, rest and posture, which are then stored separately in two `ArrayList` class variables, `restData` and `postureData`. When an external sensor records hand tremor signals, they are stored in the device's internal memory. These signals must be exported to the application and assigned to appropriate variables, namely `restData` and `postureData`. After recording and saving both positions, the two lists are merged into one named `tremorData`. Finally, a new JSONArray variable called `dataArray` is created from the merged list. This JSON array is

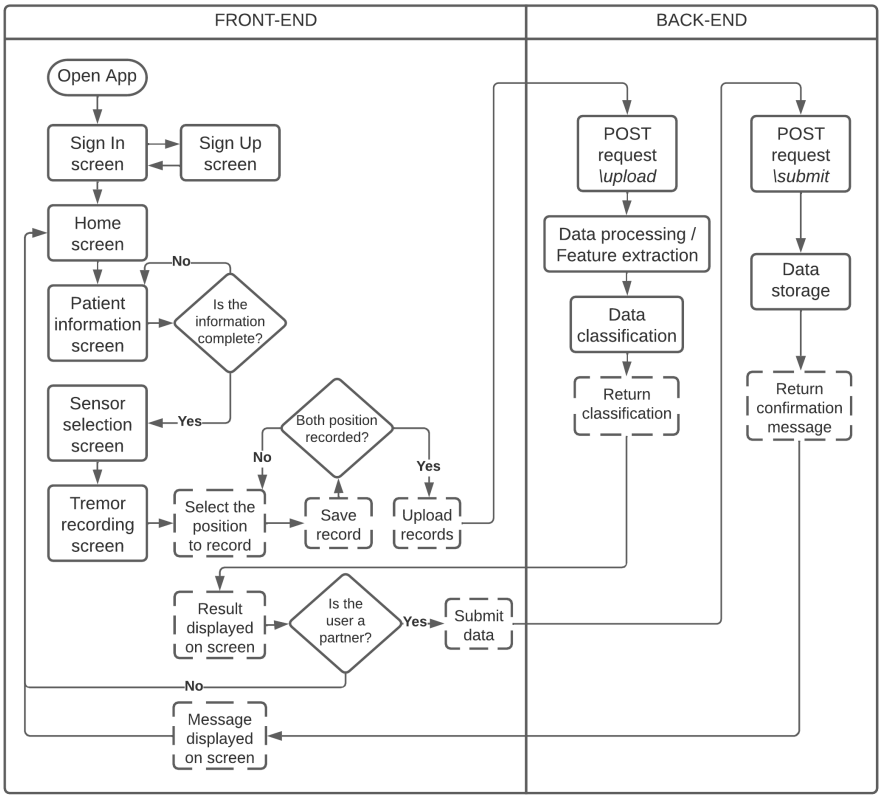



Figure 3.3: Flowchart of the TremorSoft tool.

sent to the server hosted on the Heroku platform, where the signals undergo processing, analysis, and classification. Subsequently, the application receives and presents a message corresponding to the outcome provided by the server. Depending on the outcome and classification achieved, the application can display one of three messages: 1) A pathological tremor, classified as Parkinson's, has been detected; 2) A pathological tremor, classified as an Essential tremor, has been detected. 3) The recorded tremor has been classified as a physiological tremor.

The SEND button is enabled if the application user is an accredited TremorSoft collaborator. Once the user receives the classification, they confirm it by pressing the button and authorising the sending and storage of recorded data in the web server's database. When the process is initiated, the clinical and diagnostic data are collected and stored in a new JSONObject named *patientDataJSON*. Later, *patientDataJSON* is added to the JSONArray named *tremorsoftData*, which is a replica of *dataArray* that contains the signals captured during rest and posture. Finally, *tremorsoftData* is transferred via a POST request using the path */submit*. Finally, upon completing the task, the server returns a confirmation message on the application's screen. The user can then choose to either restart the classification process on the same patient using the "RESET" button or perform a new test on a different patient by selecting the "NEW" button. Please refer to Figure 3.4 for the

software's user interface.

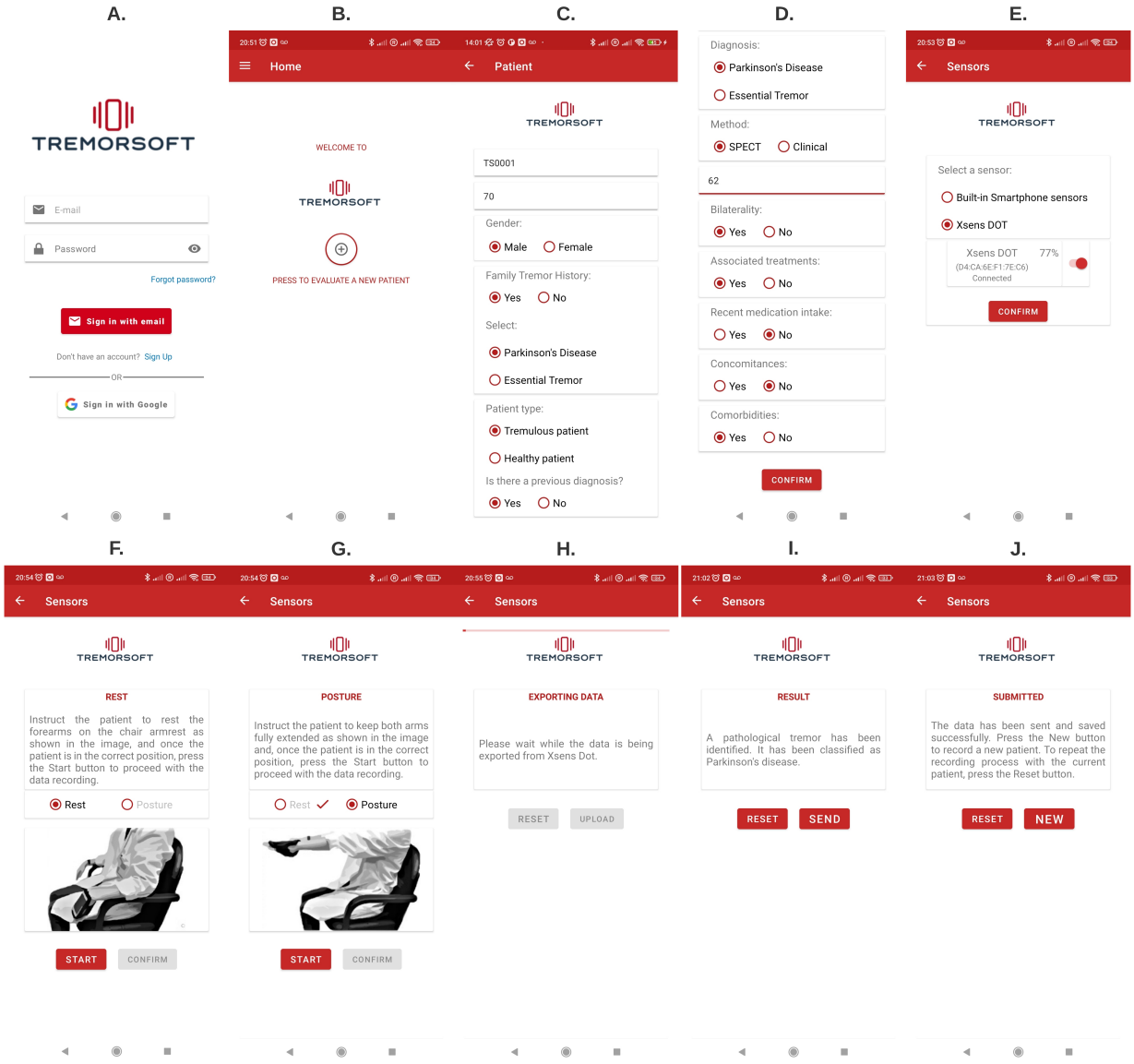



Figure 3.4: Overview of the dialogue screens of the TremorSoft user interface: A) Login screen; B) Home screen; C) Basic patient data; D) Patient diagnostic data; E) Sensor selection for recording hand tremor; F) Recording hand tremor at rest; G) Recording hand tremor in the posture position; H) Exporting records when using Xsens DOT; I) Classification result of tremor returned by the web server; J) Confirmation of data submission and storage on the web server.

### Back-end

The backend is created as a RESTful API using the Python programming language and the Flask framework, deployed on the Heroku platform. It comprises all the necessary features and components for analyzing, processing, and categorizing hand tremor records and storing them and other

patient information in the SQL database. The following sections provide detailed descriptions of the three primary components of the TremorSoft web server.

1. **Data Preprocessing and Kinematic Feature Extraction:** The server receives the JSONArray through the " */upload*" route from the mobile application and converts it into a Pandas DataFrame for further processing. Initially, the signals undergo preprocess filtering to decrease sensor drift and distortions caused by various physical phenomena. Then, the Power Spectral Density (PSD) of each accelerometer and gyroscope axis is calculated. Kinematic features are extracted from the components' power spectral density (PSD) and then assessed through machine learning models.

2. **Hand Tremor Classification Using Machine Learning:** The server stores the classification models that were created based on methodology and results obtained from previous studies [4, 94, 95]. In the classification process, a specific model and kinematic features previously extracted through the preprocessing function are utilized to classify the recorded tremor as either physiological or pathological initially. If the first model classifies the tremor as pathological, a second model is utilized to distinguish if the subject's tremor is Parkinsonian or Essential Tremor. Despite several kinematic features extracted in prior research, the classification models solely utilize a limited number of features, particularly those that offer high predictive power in combination with the model.

3. **Storage of Patient Data with Confirmed Diagnosis:** An SQL database is connected to the server via the Heroku Postgres service to upload and store data. Once all the necessary information is verified, subject data is uploaded to the web server through a POST request on the /upload route. The dataset stored for each patient consists of 25 columns, as illustrated in Figures 3.5 and 3.6. Patient identification ($Patient_id$), age, gender, family history of tremor ($F_history$), machine learning-based classification (Classification), prior diagnosis (Diagnosis), diagnostic method (Method), age of disease onset/diagnosis ($O_age$), bilaterality, and received treatments (Treatments), Medication taken prior to imaging (Medication), Comorbidities, Acceleration signals at rest ($AccX_r$, $AccY_r$, $AccZ_r$), Acceleration signals in posture ($AccX_p$, acceleration signals at rest ($AccX_r$, $AccY_r$, $AccZ_r$), acceleration signals in posture ($AccX_p$, $AccY_p$, $AccZ_p$), angular velocity signals at rest ($GyrX_r$, $GyrY_r$, $GyrZ_r$), and angular velocity signals in posture ($GyrX_p$, $GyrY_p$, $GyrZ_p$).

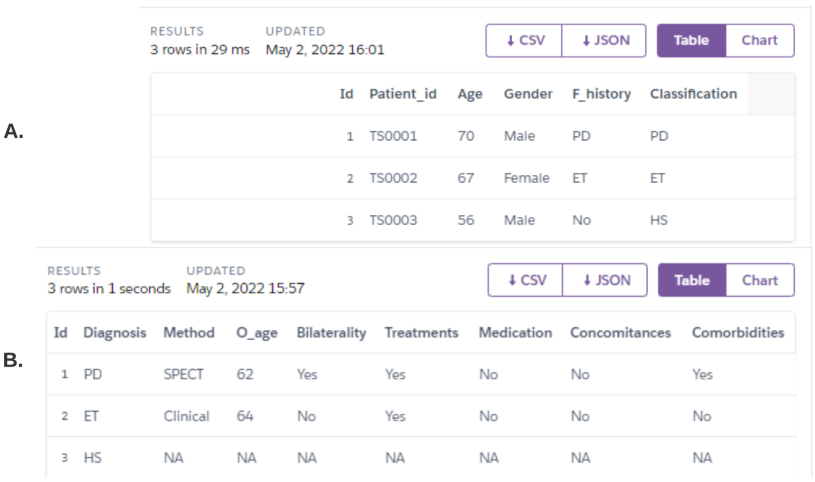



Figure 3.5: Patient data stored in the SQL database hosted on the Heroku server: A. Basic patient data and tremor classification; B. Diagnosis-related data.

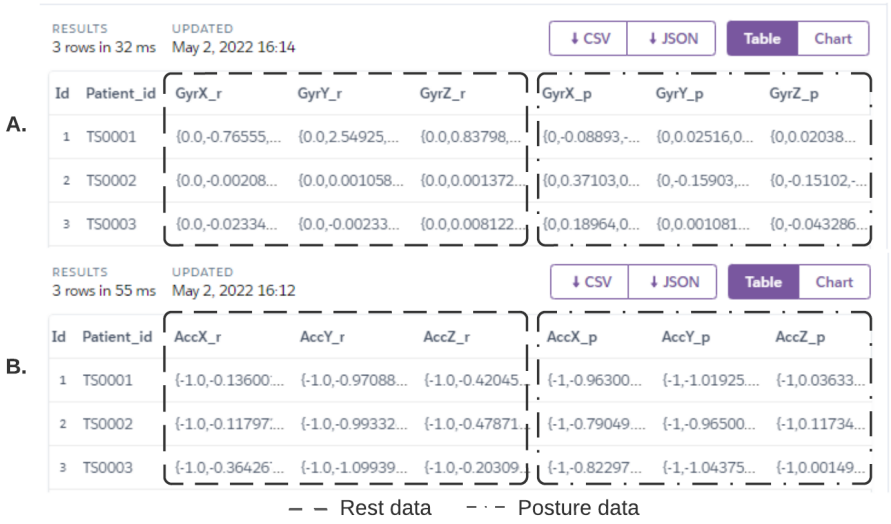



Figure 3.6: Data from the resting and postural tremor records stored in the SQL database hosted on the Heroku server: A) Records of angular velocities from the gyroscope; B) Records of linear accelerations from the accelerometer.

## 3.5 Impact

The medical significance of this e-health application is to achieve further progression and knowledge beyond the current state-of-the-art testing methods for movement disorders. This decision-support tool functions as an added evaluation technique to distinguish pathological tremors, which can be challenging to identify in certain cases, particularly during the initial stages of the disease. Furthermore, this application will achieve a high level of reliability in assisting neurologists with accurate evaluation and identification of movement disorders, as well as measuring their severity. The knowl-

edge generated from this tool will represent a substantial scientific contribution to improving the differential diagnosis of various movement disorders when compared to the information obtained from SPECT.

This application aims to provide objective information to facilitate decision-making and reduce wait times for a final diagnosis, enabling patients to receive appropriate treatment quickly. The application's classification cannot be considered a definitive diagnosis, but it enhances decision-making for doctors and specialists in movement disorders who perform assessments. These assessments consider additional clinical criteria in addition to hand tremors. Likewise, if there is suspicion of a false positive or false negative due to an error in patient registration, the treating physician can repeat the test as patient registration and classification are completed quickly.

In future endeavors, the goal is to seek collaboration with various movement disorder centers to expand the database with records of patients with confirmed diagnoses, to continually improve and maintain the implemented models, and thus to have a higher degree of reliability in the classifications performed by the models. Considering the nature of the data collected from both users (name, email, profile picture, etc.) and patients (age, gender, diagnosis, etc.), it will be ensured that the final version of TremorSoft complies with all the standards and measures imposed by the General Data Protection Regulation (GDPR) by encrypting personal data, preventing unauthorized access to this data, and constantly evaluating the security measures implemented. Finally, we plan to add a new feature to the mobile application that enables sending reports to the email addresses of physicians and specialists who use the app in their clinical practice.

## 3.6 Conclusions

This work provides a quantitative, user-friendly, non-invasive, and cost-effective method that can be used as a decision-support tool in diagnosing Parkinson's disease and Essential Tremor based on hand tremor recording. The tremor classification result is available quickly during the medical evaluation by the physician, either in person or remotely. The combination of clinical information with information from kinematic features for training machine learning models is the key to the functionality of this tool, providing the application with higher classification accuracy. Typically, the classification of these motor disorders focuses on obtaining one or more kinematic biomarkers; however, the heterogeneity of both diseases complicates this approach, and we believe that complementing clinical data with kinematic biomarkers is more efficient.

# Chapter 4

# General Discussion

In recent years, machine learning and deep learning fields have experienced rapid advancements, enabling their application in various medical domains that require large volumes of data or images. In neurology, particularly in the realm of movement disorders, machine learning techniques have emerged as a valuable tool for detailed characterization and precise classification of different types of tremors. Therefore, this chapter aims to discuss the importance of employing machine learning methods in characterizing tremor-related disorders and to highlight the contributions made, focusing on their role in early diagnosis, differentiation between different types of tremors, and continuous patient monitoring.

**Early diagnosis** is crucial for providing timely treatment and improving the quality of life for patients with tremor-related disorders. Traditional diagnostic methods often rely on subjective clinical evaluation by doctors based on their expertise, which can lead to errors and delays in diagnosis. Machine learning has demonstrated its ability to identify signs and symptoms of a disease before they become visible to doctors, allowing for early and accurate detection of tremor-related disorders. For instance, Wang et al. [110] used a deep learning approach to diagnose tremors with over 80% accuracy based on specific tremor features in videos of the patients. Here, the focus was on acquiring signals from the upper extremities during a short clinical visit. This is one of this technology's most important requirements and innovations, as doctors can access more sophisticated equipment with higher sensitivity, such as SPECT or EMG. The issue is that using SPECT requires waiting lists of months and patient compatibility with chemicals that need to be introduced into the bloodstream. On the other hand, EMG is a tool available in most neurology departments. Still, signal acquisition takes a long time due to equipment setup and is very sensitive to noise or vibrations. Therefore, the developed tool allows signal recording and provides an assessment within the 5 minutes the doctor has during the visit of the patient. The ease of use and the speed with

which this tool records and provides objective information based on machine learning classification models with previously recorded and validated data can assist doctors, especially in two specific scenarios. The first scenario is with patients in the early stages of the disease, where the doctor may doubt the diagnosis. The second scenario occurs in regions where more sophisticated equipment is unavailable to confirm the patient's condition.

**Differentiating between types of tremors** is essential for determining the most suitable treatment for the patient and potentially improving their quality of life. As the introduction mentions, Essential Tremor and tremor associated with Parkinson's disease have similar characteristics but require different treatment approaches. Machine learning has allowed the development of classification models that can differentiate between types of tremors with exceptional accuracy. One of the initial studies by Sigcha et al. [111] used a machine learning algorithm to classify Parkinsonian tremors based solely on wrist accelerometer data successfully. Building on this study, our work focused on including gyroscope data and various kinematic features of the signal to enhance differentiation between populations. While it is true that all studies focus on differentiating between Parkinson's and Essential Tremor because they exhibit similar motor disorders, the required treatments are very different. The importance lies in early and appropriate treatment, which significantly prolongs the quality of life for Parkinson's patients [112]. Therefore, there is no interest in comparing with other disorders with much lower incidence in the population. Within the realm of tremor differentiation, using inertial sensors and classification models to characterize the severity of movement disorders may be particularly relevant. This allows doctors to determine the best course of treatment and helps patients and their families better understand the progression of the disease.

**Continuous monitoring** of tremor is essential for assessing disease progression and adjusting treatment as needed. It also allows us to study the acceptance of different medications and their side effects in patients, a standard line of work in many movement disorder departments [112]. Wireless technologies, such as inertial sensors that integrate accelerometers and gyroscopes, have made it easier to collect real-time tremor data. These devices are advantageous because they are small devices, easy to place and use. However, manual analysis of this data is labor-intensive and error-prone. Machine learning provides a solution by processing large volumes of tremor data and extrapolating meaningful patterns. For example, Nilashi et al. [113] used a machine-learning approach to predict the progression of Parkinson's disease using data from an inertial sensor, allowing continuous and non-invasive monitoring. In our case, to create an integrated and user-friendly tool for the end user, a mobile app is developed that combines three primary functions: recording, storage, and processing. In the first recording activity, a wireless inertial sensor continuously

records the signal for the desired frequency within the limits of the sensors. Second, the recorded signals are stored in a cloud database, eliminating the need for local desktop storage and the time required to organize information correctly. Finally, signal processing is performed on the server using an optimized classification model that compares the new signal with the database, contrasted with patients previously diagnosed and validated by more sophisticated equipment. Therefore, it is believed that there is room for improvement in making this tool more robust and refined, but it represents progress in patient monitoring because it does not require great complexity and time for use.

**Rehabilitation** with inertial sensors is a technique that can help patients with tremor disease improve their balance, coordination, and mobility [114]. Inertial sensors can be used to provide feedback on gait and movements, which can help patients become more aware of their deficits and perform more efficient movements. Rehabilitation with inertial sensors can be performed in a clinical or home environment. Therapists can use inertial sensors to design personalized exercises for each patient in a clinical setting. Patients can use inertial sensors to perform activities independently in a home environment. Several studies have shown that rehabilitation with inertial sensors can effectively improve Parkinson's disease symptoms [115]. For example, one study found that Parkinson's patients who underwent rehabilitation with inertial sensors for 12 weeks significantly improved their balance, coordination, and walking speed. While we have not had the opportunity to use the tool or developed models in this area, the possibility remains open for recording repetitive movements and studying how signal recordings evolve over time.

## 4.1 Main limitations

Machine learning models have proven decisive in differentiating between Parkinson's disease and Essential Tremor patients, but certain limitations must be stressed. First, their performance and generalization heavily depend on the quality and quantity of training data and the appropriate selection of parameters. Using a low number of patients and an excess of parameters can lead to several limitations and issues affecting the effectiveness and reliability of the models. Accordingly, the main limitations of this work are discussed in detail below.

With a limited number of patients compared to the complexity of the model, there is a significant risk of overfitting. Overfitting occurs when the model fits too closely to noisy training data and does not generalize well to new data. This can lead to deceptively good results on the training data but poor performance on unseen data. Additionally, a low number of patients may fail to capture the diversity and variability present in the target population. This can result in biases

and poor generalization to demographic groups not represented in the dataset. The effectiveness of the model can be highly sensitive to slight variations in the training set due to the lack of sufficient data to provide a robust and stable view of patterns. To address these issues and given that the study initially started with a limited patient base, over the years, it has been expanded to include 98 registered subjects, comprising 49 Parkinson's patients, 27 with Essential Tremor, and 22 healthy individuals. Although it remains a relatively modest database, iterative methods combining different proportions of training and validation patients have been used to limit the effect of the relatively low number of patients. The future goal is to continue expanding the database through three new collaboration agreements with Asociación Catalana para el Parkinson, Consorci Sanitari Alt Penedès-Garraf and the ONG Chilena de Centro de Trastornos del Movimiento (CETRAM), as outlined in Appendix I.

Another issue faced throughout this work is the volume of parameters used. The adopted procedure can be observed in the articles published regarding the studied sensor and its details. Initially, the work started with the mean values of the accelerometer signals, eventually delving into the contribution of each component of the triad to the accelerometer and gyroscope signals. This led to the generation of complex models with a large number of parameters, which introduced the following problems:

1. An excess of parameters in a model can increase its complexity and ability to fit training data, leading to overfitting. These models negatively impact their ability to generalize.

2. Models with many parameters require more computational resources for training and validation. This can increase training times and limit the scalability of the model in practical applications.

3. As the number of parameters increases, it becomes more challenging to interpret how the model makes decisions. Complex models may lack transparency and applicability, which is particularly problematic in medical applications where decision-making must be understandable to healthcare professionals.

Before developing the tool in the form of a mobile application plus web server, the project was at this stage, with several models that could classify different studied tremor populations well, and these classification models were nourished with a large number of parameters, such as the frequency proportion, kinematic features, proportion of training and validation patients, or the number of iterations to reorder the data. Consequently, various measures were adopted to mitigate these limitations to create a lightweight and useful tool. Some of the decisions made were as follows:

1. Patient training and validation data were combined to avoid overfitting and bias problems. Larger and more diverse datasets improve model generalization and robustness despite the limitation of having a small database.

2. Instead of using all developed kinematic features, only the relevant and physically significant features for the specific study problem were selected. This reduced the model's complexity, decreased the risk of overfitting, and shortened the classification process.

3. Cross-validation was used to evaluate the model's performance on different training and validation datasets, providing a more reliable estimation of overall model performance.

4. Finally, models of easy implementation were chosen over extremely complex ones. This helped facilitate model integration into the database and simplified data interpretation, even though some advantages in population differentiation might have been sacrificed.

# Chapter 5

# General Conclusions

The general conclusions are detailed in the order of accomplishment throughout the project aimed at creating a diagnostic support tool for some types of movement disorders, specifically Parkinson's disease and Essential Tremor.

1. First and foremost, a methodology for easy and rapid implementation with inertial sensors during clinical visits has been developed, allowing the recording of tremors in patients with motor disorders. Thanks to the signal recording in various positions with an inertial sensor that builts-in accelerometer and gyroscope, different kinematic features have been developed to differentiate between movement disorder types with hand tremors, regardless of the severity of the disease.

2. Secondly, machine learning classification models have enabled the identification of the optimal combination of kinematic features, the portion of the recorded signal, the training/validation signal ratio, and classification models for distinguishing between tremor disease types in two phases. In the first phase, the classification was performed between patients with tremors and healthy subjects. In the second phase, differentiation was made between patients with tremors, Parkinson's disease, and Essential Tremors.

3. Finally, the first version of TremorSoft has been developed in the form of a mobile app with a web server. This tool has three primary functions. The first is to record the patient's medical history, including any associated issues or current medication. The second function is to record the signal in desired positions on the web server. For this purpose, an interactive tutorial is available, guiding the doctor/patient on the positions to be assumed for adequately recording the signal using the smartphone or wireless inertial sensors. The last function of the tool is to provide feedback to the doctor regarding the type of population to which the registered subject

belongs based on the classification made by the machine learning model implemented on the web server. Additionally, collaborating doctors in the project can validate the obtained results if confirmed results are available from other more sophisticated and sensitive equipment, such as SPECT. This allows the web server to increase the base of correctly diagnosed patients from which the machine learning classification model derives its knowledge.

# Chapter 6

# Future Work

This work does not conclude with this thesis. Still, it will continue in the future at this institution as the coordinator and in collaboration with various institutions such as the University of Lisbon, the University of Santiago de Chile, the ONG of Movement Disorders Center (CETRAM) of Chile, Associació Catalana per al Parkinson, and recently, Consorci Sanitari Alt Penedès-Garraf. Some points considered pending and will be the subject of future studies include the following topics.

1. First and foremost, the tool must be validated and further developed, making the app more robust and feature-rich. This requires expanding the database of registered and confirmed patients with a diagnosis. To achieve this, a framework agreement has been reached with Consorci Sanitari Alt Penedès-Garraf, which will allow us to register more patients with motor disorders, mainly Parkinson's disease and Essential Tremor patients, over the next 2 to 3 years in different hospitals and clinics in the Garraf region.

2. Improvement of the functionalities of the tool in several aspects is needed. One line of work is enabling doctors to create clinical records during the clinical visit using the tool. Additionally, there is an intention to generate clinical reports directly within the app for sending by email in PDF or RAR format for storage in the patient's medical record without the need for manual entry. Furthermore, there should be an enhancement in the ability to choose the recording frequency and recording intervals, which are currently fixed by default.

3. Optimization of the classification models to reduce analysis times by using the minimum number of parameters (kinematic features, sensor axis, frequency range, etc.) necessary to achieve high accuracy in differentiating between patients with tremors. This optimization offers other associated advantages, such as the ease of interpreting the results and understanding the physical behavior with the parameters involved in the model. Another advantage is that

the simpler the classification model is implemented on the cloud server, the less computation time is required, resulting in a shorter response time for the doctor to access the necessary information without delay.

4. The study of the impact of medication cycles on the mobility of Parkinson's patients. Since there are different medications, Parkinson's patients must work with their doctors to find the combination of drugs and doses that work best for them. Therefore, the developed technology allows adaptation for patient monitoring over the 24 hours required to study these medication cycles.

# References


[1] European Commission. *Horizon 2020*. Nov. 2013. URL: https://ec.europa.eu/programmes/horizon2020/en/.

[2] European Commission / Directorate-General for Research and Innovation. *Horizon Europe*. Nov. 2018. URL: https://ec.europa.eu/info/horizon-europe_en.

[3] World Health Organization (WHO). *Noncommunicable diseases*. Nov. 2018. URL: https://www.who.int/news-room/fact-sheets/detail/noncommunicable-diseases.

[4] Sergi Barrantes et al. "Differential diagnosis between Parkinson's disease and essential tremor using the smartphone's accelerometer". In: *PLOS ONE* 12.8 (Aug. 2017). Ed. by Robert Chen, e0183843. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0183843. URL: https://dx.plos.org/10.1371/journal.pone.0183843.

[5] Patrick Locatelli and Dario Alimonti. "Differentiating essential tremor and Parkinson's disease using a wearable sensor  A pilot study". In: *2017 7th IEEE International Workshop on Advances in Sensors and Interfaces (IWASI)*. IEEE, June 2017, pp. 213–218. ISBN: 978-1-5090-6707-7. DOI: 10.1109/IWASI.2017.7974254. URL: http://ieeexplore.ieee.org/document/7974254/.

[6] E. Ray Dorsey et al. "Global, regional, and national burden of Parkinson's disease, 19902016: a systematic analysis for the Global Burden of Disease Study 2016". In: *The Lancet Neurology* 17.11 (Nov. 2018), pp. 939–953. ISSN: 14744422. DOI: 10.1016/S1474-4422(18)30295-3. URL: /pmc/articles/PMC6191528/?report=abstract%20https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6191528/%20https://linkinghub.elsevier.com/retrieve/pii/S1474442218302953.

[7] Lorraine N Clark and Elan D Louis. "Essential tremor". In: *Handbook of Clinical Neurology*. Vol. 147. 2018, pp. 229–239. ISBN: 9780444632333. DOI: 10.1016/B978-0-444-63233-3.00015-4. URL: https://linkinghub.elsevier.com/retrieve/pii/B9780444632333000154.

[8] Diane B. Miller and James P. OCallaghan. “Biomarkers of Parkinsons disease: Present and future”. In: *Metabolism* 64.3 (Mar. 2015), S40–S46. ISSN: 00260495. DOI: 10.1016/j.metabol.2014.10.030. URL: http://dx.doi.org/10.1016/j.metabol.2014.10.030%20https://linkinghub.elsevier.com/retrieve/pii/S0026049514003394.

[9] Santosh Kumar Nanda et al. “A quantitative classification of essential and Parkinson’s tremor using wavelet transform and artificial neural network on sEMG and accelerometer signals”. In: *2015 IEEE 12th International Conference on Networking, Sensing and Control*. IEEE, Apr. 2015, pp. 399–404. ISBN: 978-1-4799-8069-7. DOI: 10.1109/ICNSC.2015.7116070. URL: http://ieeexplore.ieee.org/document/7116070/.

[10] Decho Surangsrirat et al. “Support vector machine classification of Parkinson’s disease and essential tremor subjects based on temporal fluctuation”. In: *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, Aug. 2016, pp. 6389–6392. ISBN: 978-1-4577-0220-4. DOI: 10.1109/EMBC.2016.7592190. URL: http://ieeexplore.ieee.org/document/7592190/.

[11] Nin Bajaj, Robert A. Hauser, and Igor D. Grachev. “Clinical utility of dopamine transporter single photon emission CT (DaT-SPECT) with ( 123 I) ioflupane in diagnosis of parkinsonian syndromes”. In: *Journal of Neurology, Neurosurgery & Psychiatry* 84.11 (Nov. 2013), pp. 1288–1295. ISSN: 0022-3050. DOI: 10.1136/jnnp-2012-304436. URL: https://jnnp.bmj.com/lookup/doi/10.1136/jnnp-2012-304436%20http://www.ncbi.nlm.nih.gov/pubmed/23486993%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3812862.

[12] Lazzaro di Biase et al. “Tremor stability index: a new tool for differential diagnosis in tremor syndromes”. In: *Brain* 140.7 (July 2017), pp. 1977–1986. ISSN: 0006-8950. DOI: 10.1093/brain/awx104. URL: https://academic.oup.com/brain/article/140/7/1977/3782569%20http://www.ncbi.nlm.nih.gov/pubmed/28459950%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC5493195.

[13] Fernando L. Pagan. “Improving outcomes through early diagnosis of Parkinson’s disease.” In: *The American journal of managed care* 18.7 Suppl (Sept. 2012), pp. 176–82. ISSN: 1936-2692. URL: http://www.ncbi.nlm.nih.gov/pubmed/23039866.

[14] Warwick R. Adams. “High-accuracy detection of early Parkinson’s Disease using multiple characteristics of finger movement while typing”. In: *PLOS ONE* 12.11 (Nov. 2017). Ed. by Holger Fröhlich, e0188226. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0188226. URL: https://dx.plos.org/10.1371/journal.pone.0188226.

[15] Kei Uchida et al. “Tremor is attenuated during walking in essential tremor with resting tremor but not parkinsonian tremor”. In: *Journal of Clinical Neuroscience* 18.9 (Sept. 2011), pp. 1224–1228. ISSN: 09675868. DOI: 10.1016/j.jocn.2010.12.053. URL: https://linkinghub.elsevier.com/retrieve/pii/S096758681100169X.

[16] Musleh Algarni and Alfonso Fasano. “The overlap between Essential tremor and Parkinson disease”. In: *Parkinsonism & Related Disorders* 46 (Jan. 2018), S101–S104. ISSN: 13538020. DOI: 10.1016/j.parkreldis.2017.07.006. URL: https://doi.org/10.1016/j.parkreldis.2017.07.006%20https://linkinghub.elsevier.com/retrieve/pii/S1353802017302390.

[17] Frank Papengut et al. “Rest tremor suppression may separate essential from parkinsonian rest tremor”. In: *Parkinsonism & Related Disorders* 19.7 (July 2013), pp. 693–697. ISSN: 13538020. DOI: 10.1016/j.parkreldis.2013.03.013. URL: http://dx.doi.org/10.1016/j.parkreldis.2013.03.013%20https://linkinghub.elsevier.com/retrieve/pii/S1353802013001296.

[18] A H Hristova and W C Koller. “Early Parkinson’s disease: what is the best approach to treatment.” In: *Drugs & aging* 17.3 (Sept. 2000), pp. 165–81. ISSN: 1170-229X. DOI: 10.2165/00002512-200017030-00002. URL: http://www.ncbi.nlm.nih.gov/pubmed/11043817.

[19] Hani T. S. Benamer et al. “Accurate differentiation of parkinsonism and essential tremor using visual assessment of [123I]-FP-CIT SPECT imaging: The [123I]-FP-CIT study group”. In: *Movement Disorders* 15.3 (May 2000), pp. 503–510. ISSN: 0885-3185. DOI: 10.1002/1531-8257(200005)15:3<503::AID-MDS1013>3.0.CO;2-V. URL: https://onlinelibrary.wiley.com/doi/10.1002/1531-8257(200005)15:3%3C503::AID-MDS1013%3E3.0.CO;2-V.

[20] Jacques Darcourt et al. “EANM procedure guidelines for brain neurotransmission SPECT using (123)I-labelled dopamine transporter ligands, version 2.” In: *European journal of nuclear medicine and molecular imaging* 37.2 (Feb. 2010), pp. 443–50. ISSN: 1619-7089. DOI: 10.1007/s00259-009-1267-x. URL: http://www.ncbi.nlm.nih.gov/pubmed/19838702.

[21] Roger N. Rosenberg and Juan M. Pascual. *Rosenberg’s Molecular and Genetic Basis of Neurological and Psychiatric Disease*. Elsevier, 2015, pp. 1–1424. ISBN: 9780124105294. DOI: 10.1016/C2012-0-02688-9. URL: https://linkinghub.elsevier.com/retrieve/pii/C20120026889.

[22] Kailash P. Bhatia et al. “Consensus Statement on the classification of tremors. from the task force on tremor of the International Parkinson and Movement Disorder Society”. In: *Movement Disorders* 33.1 (Jan. 2018), pp. 75–87. ISSN: 08853185. DOI: 10.1002/mds.27121. URL: http://doi.wiley.com/10.1002/mds.27121.

[23] C Bhavana et al. “Techniques of measurement for Parkinson’s tremor highlighting advantages of embedded IMU over EMG”. In: *2016 International Conference on Recent Trends in Information Technology (ICRTIT)*. IEEE, Apr. 2016, pp. 1–5. ISBN: 978-1-4673-9802-2. DOI: 10.1109/ICRTIT.2016.7569560. URL: http://ieeexplore.ieee.org/document/7569560/.

[24] Alan Michael Woods et al. “Parkinsons disease and essential tremor classification on mobile device”. In: *Pervasive and Mobile Computing* 13 (Aug. 2014), pp. 1–12. ISSN: 15741192. DOI: 10.1016/j.pmcj.2013.10.002. URL: http://dx.doi.org/10.1016/j.pmcj.2013.10.002%20https://linkinghub.elsevier.com/retrieve/pii/S1574119213001260.

[25] Chandrabhaga Miskin and Karen S. Carvalho. “Tremors: Essential Tremor and Beyond”. In: *Seminars in Pediatric Neurology* 25 (Apr. 2018), pp. 34–41. ISSN: 10719091. DOI: 10.1016/j.spen.2018.02.002. URL: https://linkinghub.elsevier.com/retrieve/pii/S1071909118300044.

[26] Elan D. Louis. “Essential tremor then and now: How views of the most common tremor diathesis have changed over time”. In: *Parkinsonism & Related Disorders* 46 (Jan. 2018), S70–S74. ISSN: 13538020. DOI: 10.1016/j.parkreldis.2017.07.010. URL: https://doi.org/10.1016/j.parkreldis.2017.07.010%20https://linkinghub.elsevier.com/retrieve/pii/S1353802017302572.

[27] Timothy R Mhyre et al. “Parkinson’s disease.” In: *Sub-cellular biochemistry* 65 (2012), pp. 389–455. ISSN: 0306-0225. DOI: 10.1007/978-94-007-5416-4{_}16. URL: http://link.springer.com/10.1007/978-94-007-5416-4_16%20http://www.ncbi.nlm.nih.gov/pubmed/23225012%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4372387.

[28] M.F. Lew and Y. Yeung. “Parkinson’s Disease”. In: *Reference Module in Biomedical Sciences*. Elsevier, Jan. 2014. DOI: 10.1016/B978-0-12-801238-3.00146-X. URL: https://linkinghub.elsevier.com/retrieve/pii/B978012801238300146X.

[29] Stephanie Janezic et al. “Deficits in dopaminergic transmission precede neuron loss and dysfunction in a new Parkinson model”. In: *Proceedings of the National Academy of Sciences of the United States of America* 110.42 (Oct. 2013), E4016–E4025. ISSN: 00278424. DOI: 10.1073/pnas.1309143110. URL: http://www.pnas.org/cgi/doi/10.1073/pnas.1309143110%20http://www.ncbi.nlm.nih.gov/pubmed/24082145%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3801069.

[30] Javier Blesa and Serge Przedborski. “Parkinsons disease: Animal models and dopaminergic cell vulnerability”. In: *Frontiers in Neuroanatomy* 8.DEC (2014). ISSN: 16625129. DOI: 10.3389/fnana.2014.00155.

[31] Daniel S Marín et al. “Enfermedad de Parkinson: fisiopatología, diagnóstico y tratamiento”. In: *Revista de la Universidad Industrial de Santander. Salud* 50.1 (Mar. 2018), pp. 79–92. ISSN: 01210807. DOI: 10.18273/revsal.v50n1-2018008. URL: http://revistas.uis.edu.co/index.php/revistasaluduis/article/view/8004/8155.

[32] Bryan A. Killinger et al. “The vermiform appendix impacts the risk of developing Parkinson’s disease.” In: *Science translational medicine* 10.465 (Oct. 2018), eaar5280. ISSN: 1946-6242. DOI: 10.1126/scitranslmed.aar5280. URL: https://stm.sciencemag.org/lookup/doi/10.1126/scitranslmed.aar5280%20http://www.ncbi.nlm.nih.gov/pubmed/30381408%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC6319259.

[33] Susanne Fonseca Santos et al. “The Gut and Parkinson’s DiseaseA Bidirectional Pathway”. In: *Frontiers in Neurology* 10.JUN (June 2019). ISSN: 1664-2295. DOI: 10.3389/fneur.2019.00574. URL: https://www.frontiersin.org/article/10.3389/fneur.2019.00574/full.

[34] Alison Yarnall, Neil Archibald, and David Burn. “Parkinson’s disease”. In: *Medicine* 40.10 (Oct. 2012), pp. 529–535. ISSN: 13573039. DOI: 10.1016/j.mpmed.2012.07.008. URL: http://dx.doi.org/10.1016/j.mpmed.2012.07.008%20https://linkinghub.elsevier.com/retrieve/pii/S1357303912001624.

[35] Octavio Martinez Manzanera et al. “Tremor Detection Using Parametric and Non-Parametric Spectral Estimation Methods: A Comparison with Clinical Assessment”. In: *PLOS ONE* 11.6 (June 2016). Ed. by Mikhail A. Lebedev, e0156822. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0156822. URL: https://dx.plos.org/10.1371/journal.pone.0156822.

[36] Alessandro Tessitore et al. “Sensorimotor connectivity in Parkinson’s disease: The role of functional neuroimaging”. In: *Frontiers in Neurology* 5.SEP (Sept. 2014). ISSN: 16642295. DOI: 10.3389/fneur.2014.00180. URL: http://journal.frontiersin.org/article/10.3389/fneur.2014.00180/abstract.

[37] Stephen G. Reich. “Diagnosing Parkinsons Disease”. In: *Current Clinical Neurology*. 2019, pp. 3–6. DOI: 10.1007/978-3-319-97897-0{_}1. URL: http://link.springer.com/10.1007/978-3-319-97897-0_1.

[38] Enrico Saracchi, Silvia Fermi, and Laura Brighina. “Emerging candidate biomarkers for Parkinson’s disease: a review.” In: *Aging and disease* 5.1 (Feb. 2014), pp. 27–34. ISSN: 2152-5250. DOI: 10.14366/AD.2014.050027. URL: http://www.ncbi.nlm.nih.gov/

pubmed/24490114%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3901611.

[39] Raúl Martínez-Fernández. et al. "ACTUALIZACIÓN EN LA ENFERMEDAD DE PARKINSON". In: *Revista Médica Clínica Las Condes* 27.3 (May 2016), pp. 363–379. ISSN: 07168640. DOI: 10.1016/j.rmclc.2016.06.010. URL: http://dx.doi.org/10.1016/j.rmclc.2016.06.010%20https://linkinghub.elsevier.com/retrieve/pii/S0716864016300372.

[40] Franziska Hopfner and Rick C. Helmich. "The etiology of essential tremor: Genes versus environment". In: *Parkinsonism & Related Disorders* 46 (Jan. 2018), S92–S96. ISSN: 13538020. DOI: 10.1016/j.parkreldis.2017.07.014. URL: https://doi.org/10.1016/j.parkreldis.2017.07.014%20https://linkinghub.elsevier.com/retrieve/pii/S1353802017302614.

[41] Daryl J. Wile, Ranjit Ranawaya, and Zelma H.T. Kiss. "Smart watch accelerometry for analysis and diagnosis of tremor". In: *Journal of Neuroscience Methods* 230 (June 2014), pp. 1–4. ISSN: 01650270. DOI: 10.1016/j.jneumeth.2014.04.021. URL: http://dx.doi.org/10.1016/j.jneumeth.2014.04.021%20https://linkinghub.elsevier.com/retrieve/pii/S016502701400137X.

[42] Kyum-Yil Kwon et al. "Comparison of motor and non-motor features between essential tremor and tremor dominant Parkinson's disease". In: *Journal of the Neurological Sciences* 361 (Feb. 2016), pp. 34–38. ISSN: 0022510X. DOI: 10.1016/j.jns.2015.12.016. URL: https://linkinghub.elsevier.com/retrieve/pii/S0022510X15300824.

[43] Wei Chen et al. "Topography of essential tremor". In: *Parkinsonism & Related Disorders* 40 (July 2017), pp. 58–63. ISSN: 13538020. DOI: 10.1016/j.parkreldis.2017.04.012. URL: https://linkinghub.elsevier.com/retrieve/pii/S135380201730144X.

[44] Günther Deuschl et al. "Treatment of patients with essential tremor". In: *The Lancet Neurology* 10.2 (Feb. 2011), pp. 148–161. ISSN: 14744422. DOI: 10.1016/S1474-4422(10)70322-7. URL: http://dx.doi.org/10.1016/S1474-4422(10)70322-7%20https://linkinghub.elsevier.com/retrieve/pii/S1474442210703227.

[45] Elan D. Louis. "The evolving definition of essential tremor: What are we dealing with?" In: *Parkinsonism & Related Disorders* 46 (Jan. 2018), S87–S91. ISSN: 13538020. DOI: 10.1016/j.parkreldis.2017.07.004. URL: https://doi.org/10.1016/j.parkreldis.2017.07.004%20https://linkinghub.elsevier.com/retrieve/pii/S1353802017302377.

[46] Arjun Tarakad and Joseph Jankovic. "Essential Tremor and Parkinsons Disease: Exploring the Relationship". In: *Tremor and Other Hyperkinetic Movements* 8.0 (Jan. 2019), p. 589.

ISSN: 2160-8288. DOI: 10.5334/tohm.441. URL: https://tremorjournal.org/article/10.5334/tohm.441/.

[47] Thomas Mitchell. *Machine Learning*. McGraw-Hill, 1997, p. 414. ISBN: 9780071154673.

[48] Christopher M Bishop. *Pattern Recognition and Machine Learning*. 1st ed. Springer New York, NY, Aug. 2006, p. 738. ISBN: 9780387310732. URL: https://link.springer.com/book/9780387310732%20papers2://publication/uuid/05A8B4CF-0248-4692-8B1D-DCC065B79465.

[49] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *Nature* 521.7553 (May 2015), pp. 436–444. ISSN: 0028-0836. DOI: 10.1038/nature14539. URL: https://www.nature.com/articles/nature14539.

[50] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. New York, NY: Springer New York, 2009, p. 745. ISBN: 978-0-387-84857-0. DOI: 10.1007/978-0-387-84858-7. URL: http://link.springer.com/10.1007/978-0-387-84858-7.

[51] Marina Sokolova and Guy Lapalme. "A systematic analysis of performance measures for classification tasks". In: *Information Processing & Management* 45.4 (July 2009), pp. 427–437. ISSN: 03064573. DOI: 10.1016/j.ipm.2009.03.002. URL: https://linkinghub.elsevier.com/retrieve/pii/S0306457309000259.

[52] David Powers. "Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation". In: *Journal of Machine Learning Technologies* 2.1 (2011), pp. 37–63. ISSN: 2229-3981. URL: https://researchnow.flinders.edu.au/en/publications/evaluation-from-precision-recall-and-f-measure-to-roc-informednes.

[53] Gareth James et al. *An Introduction to Statistical Learning*. Vol. 103. Springer Texts in Statistics 8. New York, NY: Springer New York, Feb. 2013, p. 085201. ISBN: 978-1-4614-7137-0. DOI: 10.1007/978-1-4614-7138-7. URL: http://link.springer.com/10.1007/978-1-4614-7138-7.

[54] David G. Kleinbaum and Mitchel Klein. *Logistic Regression*. Statistics for Biology and Health. New York, NY: Springer New York, 2010. ISBN: 978-1-4419-1741-6. DOI: 10.1007/978-1-4419-1742-3. URL: http://link.springer.com/10.1007/978-1-4419-1742-3.

[55] David W. Hosmer, Stanley Lemeshow, and Rodney X. Sturdivant. *Applied Logistic Regression*. Wiley, Mar. 2013. ISBN: 9780470582473. DOI: 10.1002/9781118548387.

[56] Corinna Cortes and Vladimir Vapnik. "Support-vector networks". In: *Machine Learning* 20.3 (Sept. 1995), pp. 273–297. ISSN: 0885-6125. DOI: 10.1007/BF00994018. URL: http://link.springer.com/10.1007/BF00994018.

[57] Nello Cristianini and John Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, Mar. 2000. ISBN: 9780521780193. DOI: 10.1017/CBO9780511801389. URL: https://www.cambridge.org/core/product/identifier/9780511801389/type/book.

[58] Bernhard Schölkopf et al. "Estimating the Support of a High-Dimensional Distribution". In: *Neural Computation* 13.7 (July 2001), pp. 1443–1471. ISSN: 0899-7667. DOI: 10.1162/089976601750264965. URL: https://direct.mit.edu/neco/article/13/7/1443-1471/6529.

[59] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, July 2008. ISBN: 9780521865715. DOI: 10.1017/CBO9780511809071. URL: https://www.cambridge.org/core/product/identifier/9780511809071/type/book.

[60] N. S. Altman. "An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression". In: *The American Statistician* 46.3 (Aug. 1992), p. 175. ISSN: 00031305. DOI: 10.2307/2685209. URL: https://www.jstor.org/stable/2685209?origin=crossref.

[61] Yoav Freund and Robert E Schapire. "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting". In: *Journal of Computer and System Sciences* 55.1 (Aug. 1997), pp. 119–139. ISSN: 00220000. DOI: 10.1006/jcss.1997.1504. URL: https://linkinghub.elsevier.com/retrieve/pii/S002200009791504X.

[62] J. R. Quinlan. "Induction of decision trees". In: *Machine Learning* 1.1 (Mar. 1986), pp. 81–106. ISSN: 0885-6125. DOI: 10.1007/BF00116251. URL: http://link.springer.com/10.1007/BF00116251.

[63] Leo Breiman et al. *Classification And Regression Trees*. Routledge, Oct. 2017. ISBN: 9781315139470. DOI: 10.1201/9781315139470. URL: https://www.taylorfrancis.com/books/9781351460491.

[64] Leo Breiman. "Random forests". In: *Machine Learning* 45.1 (Oct. 2001), pp. 5–32. ISSN: 08856125. DOI: 10.1023/A:1010933404324/METRICS. URL: https://link.springer.com/article/10.1023/A:1010933404324.

[65] Giovanni Seni and John F. Elder. *Ensemble Methods in Data Mining*. Synthesis Lectures on Data Mining and Knowledge Discovery. Cham: Springer International Publishing, 2010. ISBN: 978-3-031-00771-2. DOI: 10.1007/978-3-031-01899-2. URL: https://link.springer.com/10.1007/978-3-031-01899-2.

[66] Jerome H. Friedman. "Greedy function approximation: A gradient boosting machine." In: *The Annals of Statistics* 29.5 (Oct. 2001), pp. 1189–1232. ISSN: 0090-5364. DOI: 10.1214/aos/1013203451. URL: https://projecteuclid.org/journals/annals-of-statistics/volume-29/issue-5/Greedy-function-approximation-A-gradient-boosting-machine/10.1214/aos/1013203451.full.

[67] Tianqi Chen and Carlos Guestrin. "XGBoost". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Vol. 13-17-August-2016. New York, NY, USA: ACM, Aug. 2016, pp. 785–794. ISBN: 9781450342322. DOI: 10.1145/2939672.2939785. URL: https://dl.acm.org/doi/10.1145/2939672.2939785.

[68] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. The MIT Press, Nov. 2016, p. 800. ISBN: 9780262035613. URL: https://mitpress.mit.edu/9780262035613/deep-learning/.

[69] Seon-Min Lee et al. "Differential diagnosis of parkinsonism with visual inspection of posture and gait in the early stage". In: *Gait & Posture* 39.4 (Apr. 2014), pp. 1138–1141. ISSN: 09666362. DOI: 10.1016/j.gaitpost.2014.01.004. URL: https://linkinghub.elsevier.com/retrieve/pii/S0966636214000071.

[70] Jerry Ajay et al. "A pervasive and sensor-free Deep Learning system for Parkinsonian gait analysis". In: *2018 IEEE EMBS International Conference on Biomedical and Health Informatics, BHI 2018*. Vol. 2018-Janua. IEEE, Mar. 2018, pp. 108–111. ISBN: 9781538624050. DOI: 10.1109/BHI.2018.8333381. URL: http://ieeexplore.ieee.org/document/8333381/.

[71] Esmaeil Nikfekr et al. "Trunk movement in Parkinson's disease during rising from seated position". In: *Movement Disorders* 17.2 (Mar. 2002), pp. 274–282. ISSN: 1531-8257. DOI: 10.1002/MDS.10073. URL: https://onlinelibrary.wiley.com/doi/full/10.1002/mds.10073%20https://onlinelibrary.wiley.com/doi/abs/10.1002/mds.10073%20https://movementdisorders.onlinelibrary.wiley.com/doi/10.1002/mds.10073.

[72] Miguel Wilken et al. "Sensitivity and specificity of different hand positions to assess upper limb rest tremor". In: *Movement Disorders* 34.4 (Apr. 2019), pp. 575–579. ISSN: 08853185. DOI: 10.1002/mds.27648. URL: http://doi.wiley.com/10.1002/mds.27648%20http://www.ncbi.nlm.nih.gov/pubmed/30788866.

[73] Jean-Francois Daneault. "Could Wearable and Mobile Technology Improve the Management of Essential Tremor?" In: *Frontiers in Neurology* 9.APR (Apr. 2018). ISSN: 1664-2295. DOI: 10.3389/fneur.2018.00257. URL: http://journal.frontiersin.org/article/10.3389/fneur.2018.00257/full.

[74] Nahed Jalloul. "Wearable sensors for the monitoring of movement disorders". In: *Biomedical Journal* 41.4 (Aug. 2018), pp. 249–253. ISSN: 23194170. DOI: 10.1016/j.bj.2018.06.003. URL: https://linkinghub.elsevier.com/retrieve/pii/S2319417017304080.

[75] Asma Channa, Nirvana Popescu, and Vlad Ciobanu. "Wearable Solutions for Patients with Parkinsons Disease and Neurocognitive Disorder: A Systematic Review". In: *Sensors* 20.9 (May 2020), p. 2713. ISSN: 1424-8220. DOI: 10.3390/s20092713. URL: https://www.mdpi.com/1424-8220/20/9/2713.

[76] Bruno Bastos Godoi et al. "Parkinson's disease and wearable devices, new perspectives for a public health issue: an integrative literature review". In: *Revista da Associação Médica Brasileira* 65.11 (Nov. 2019), pp. 1413–1420. ISSN: 1806-9282. DOI: 10.1590/1806-9282.65.11.1413. URL: http://www.ncbi.nlm.nih.gov/pubmed/31800906.

[77] Felix P. Bernhard et al. "Wearables for gait and balance assessment in the neurological ward - study design and first results of a prospective cross-sectional feasibility study with 384 inpatients". In: *BMC Neurology* 18.1 (Dec. 2018), p. 114. ISSN: 1471-2377. DOI: 10.1186/s12883-018-1111-7. URL: https://bmcneurol.biomedcentral.com/articles/10.1186/s12883-018-1111-7.

[78] Julian Varghese et al. "A Smart Device System to Identify New Phenotypical Characteristics in Movement Disorders". In: *Frontiers in Neurology* 10.JAN (Jan. 2019), p. 48. ISSN: 1664-2295. DOI: 10.3389/fneur.2019.00048. URL: https://www.frontiersin.org/article/10.3389/fneur.2019.00048/full.

[79] Ken J. Kubota, Jason A. Chen, and Max A. Little. "Machine learning for large-scale wearable sensor data in Parkinson's disease: Concepts, promises, pitfalls, and futures". In: *Movement Disorders* 31.9 (Sept. 2016), pp. 1314–1326. ISSN: 08853185. DOI: 10.1002/mds.26693. URL: http://doi.wiley.com/10.1002/mds.26693.

[80] Muhammad Asad Raza et al. "Clinical decision support system for Parkinson's disease and related movement disorders". In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, Mar. 2017, pp. 1108–1112. ISBN: 978-1-5090-4117-6. DOI: 10.1109/ICASSP.2017.7952328. URL: http://ieeexplore.ieee.org/document/7952328/.

[81] G. Kramer et al. "Wavelet coherence analysis: A new approach to distinguish organic and functional tremor types". In: *Clinical Neurophysiology* 129.1 (Jan. 2018), pp. 13–20. ISSN: 13882457. DOI: 10.1016/j.clinph.2017.10.002. URL: https://doi.org/10.1016/j.clinph.2017.10.002%20https://linkinghub.elsevier.com/retrieve/pii/S1388245717310799.

[82] Mariachiara Ricci et al. “Assessment of Motor Impairments in Early Untreated Parkinson’s Disease Patients: The Wearable Electronics Impact”. In: *IEEE Journal of Biomedical and Health Informatics* 24.1 (Jan. 2020), pp. 120–130. ISSN: 2168-2194. DOI: 10.1109/JBHI.2019.2903627. URL: https://ieeexplore.ieee.org/document/8662663/.

[83] Juan Rafael Orozco-Arroyave et al. “Apkinson: the smartphone application for telemonitoring Parkinson’s patients through speech, gait and hands movement.” In: *Neurodegenerative disease management* 10.3 (June 2020), pp. 137–157. ISSN: 1758-2032. DOI: 10.2217/nmt-2019-0037. URL: http://www.ncbi.nlm.nih.gov/pubmed/32571150.

[84] Pieter L Kubben et al. “TREMOR12: An Open-Source Mobile App for Tremor Quantification”. In: *Stereotactic and Functional Neurosurgery* 94.3 (2016), pp. 182–186. ISSN: 1011-6125. DOI: 10.1159/000446610. URL: https://www.karger.com/Article/FullText/446610%20http://www.ncbi.nlm.nih.gov/pubmed/27395052.

[85] M. Linares-del Rey, L. Vela-Desojo, and R. Cano-de la Cuerda. “Mobile phone applications in Parkinson’s disease: a systematic review”. In: *Neurología (English Edition)* 34.1 (Jan. 2019), pp. 38–54. ISSN: 21735808. DOI: 10.1016/j.nrleng.2018.12.002. URL: https://linkinghub.elsevier.com/retrieve/pii/S2173580818301974.

[86] In-Uk Song et al. “Brain SPECT can differentiate between essential tremor and early-stage tremor-dominant Parkinsons disease”. In: *Journal of Clinical Neuroscience* 21.9 (Sept. 2014), pp. 1533–1537. ISSN: 09675868. DOI: 10.1016/j.jocn.2013.11.035. URL: https://pubmed.ncbi.nlm.nih.gov/24814853/%20https://linkinghub.elsevier.com/retrieve/pii/S0967586814000940%20http://www.ncbi.nlm.nih.gov/pubmed/24814853.

[87] Robert LeMoyne et al. “Implementation of an iPhone for characterizing Parkinson’s disease tremor through a wireless accelerometer application”. In: *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology*. IEEE, Aug. 2010, pp. 4954–4958. ISBN: 978-1-4244-4123-5. DOI: 10.1109/IEMBS.2010.5627240. URL: http://ieeexplore.ieee.org/document/5627240/.

[88] Jean-François Daneault et al. “Using a Smart Phone as a Standalone Platform for Detection and Monitoring of Pathological Tremors”. In: *Frontiers in Human Neuroscience* 6.DEC (Dec. 2013), pp. 1–31. ISSN: 1662-5161. DOI: 10.3389/fnhum.2012.00357. URL: http://journal.frontiersin.org/article/10.3389/fnhum.2012.00357/abstract.

[89] Denzil Ferreira, Vassilis Kostakos, and Anind K. Dey. “AWARE: Mobile Context Instrumentation Framework”. In: *Frontiers in ICT* 2.APR (Apr. 2015). ISSN: 2297-198X. DOI: 10.3389/fict.2015.00006. URL: http://journal.frontiersin.org/article/10.3389/fict.2015.00006/abstract.

[90] N. Kostikis et al. “A Smartphone-Based Tool for Assessing Parkinsonian Hand Tremor”. In: *IEEE Journal of Biomedical and Health Informatics* 19.6 (Nov. 2015), pp. 1835–1842. ISSN: 2168-2194. DOI: 10.1109/JBHI.2015.2471093. URL: http://ieeexplore.ieee.org/document/7214201/.

[91] Elina Kuosmanen et al. “Smartphone-Based Monitoring of Parkinson Disease: Quasi-Experimental Study to Quantify Hand Tremor Severity and Medication Effectiveness”. In: *JMIR mHealth and uHealth* 8.11 (Nov. 2020), e21543. ISSN: 2291-5222. DOI: 10.2196/21543. URL: http://www.ncbi.nlm.nih.gov/pubmed/33242017%20http://mhealth.jmir.org/2020/11/e21543/.

[92] Hernán A. González Rojas et al. “Time measurement characterization of stand-to-sit and sit-to-stand transitions by using a smartphone”. In: *Medical & Biological Engineering & Computing* 56.5 (May 2018), pp. 879–888. ISSN: 0140-0118. DOI: 10.1007/s11517-017-1728-5. URL: http://link.springer.com/10.1007/s11517-017-1728-5.

[93] R. García-Ramos et al. “Informe de la Fundación del Cerebro sobre el impacto social de la enfermedad de Parkinson en España”. In: *Neurología* 31.6 (July 2016), pp. 401–413. ISSN: 02134853. DOI: 10.1016/j.nrl.2013.04.008. URL: https://linkinghub.elsevier.com/retrieve/pii/S0213485313001114.

[94] Julián D. Loaiza Duque et al. “Using Machine Learning and Accelerometry Data for Differential Diagnosis of Parkinsons Disease and Essential Tremor”. In: *Communications in Computer and Information Science*. Vol. 1052. Springer, 2019, pp. 368–378. ISBN: 9783030310189. DOI: 10.1007/978-3-030-31019-6{_}32. URL: http://link.springer.com/10.1007/978-3-030-31019-6_32.

[95] Julian D. Loaiza Duque et al. “Angular Velocity Analysis Boosted by Machine Learning for Helping in the Differential Diagnosis of Parkinsons Disease and Essential Tremor”. In: *IEEE Access* 8 (2020), pp. 88866–88875. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.2993647. URL: https://ieeexplore.ieee.org/document/9091038/.

[96] Julián D. Loaiza Duque et al. “TremorSoft: An decision support application for differential diagnosis between Parkinsons disease and essential tremor”. In: *SoftwareX* 22 (May 2023), p. 101393. ISSN: 2352-7110. DOI: 10.1016/J.SOFTX.2023.101393. URL: https://www.softxjournal.com/article/S2352-7110(23)00089-4/fulltext.

[97] Maria Alejandra Ceron et al. “Characterization of the volume and thickness of DIEP flap by CTA image processing”. In: *2021 22nd Symposium on Image, Signal Processing and Artificial Vision, STSIVA 2021 - Conference Proceedings* (2021). DOI: 10.1109/STSIVA53688.2021.9592004.

[98] Erick D.Chávez Pereda et al. “Experimental Data-Driven Insertion Force Analyses of Hypodermic Needles in a Soft Tissue with an In-House Test Bench”. In: *Communications in Computer and Information Science* 1685 CCIS (2022), pp. 415–422. ISSN: 18650937. DOI: 10.1007/978-3-031-20611-5{_}34/COVER. URL: https://link.springer.com/chapter/10.1007/978-3-031-20611-5_34.

[99] Srivani Padma G et al. “Parkinsonian hand Tremor Measurement device based on Fiber Bragg Grating sensor”. In: *2015 International Conference on Smart Sensors and Systems (IC-SSS)*. IEEE, Dec. 2015, pp. 1–3. ISBN: 978-1-4673-9328-7. DOI: 10.1109/SMARTSENS.2015.7873611. URL: http://ieeexplore.ieee.org/document/7873611/.

[100] Thomas Novak and Karl M. Newell. “Physiological tremor (812Hz component) in isometric force control”. In: *Neuroscience Letters* 641 (Feb. 2017), pp. 87–93. ISSN: 03043940. DOI: 10.1016/j.neulet.2017.01.034. URL: https://www.sciencedirect.com/science/article/abs/pii/S0304394017300447%20https://linkinghub.elsevier.com/retrieve/pii/S0304394017300447.

[101] P. Welch. “The use of fast Fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms”. In: *IEEE Transactions on Audio and Electroacoustics* 15.2 (June 1967), pp. 70–73. ISSN: 0018-9278. DOI: 10.1109/TAU.1967.1161901. URL: http://ieeexplore.ieee.org/document/1161901/.

[102] Jundong Li et al. “Feature selection: A data perspective”. In: *ACM Computing Surveys* 50.6 (Jan. 2017), pp. 1–45. ISSN: 15577341. DOI: 10.1145/3136625. URL: http://arxiv.org/abs/1601.07996%20http://dx.doi.org/10.1145/3136625%20https://dl.acm.org/doi/10.1145/3136625.

[103] Utkarsh Mahadeo Khaire and R. Dhanalakshmi. “Stability of feature selection algorithm: A review”. In: *Journal of King Saud University - Computer and Information Sciences* (June 2019). ISSN: 13191578. DOI: 10.1016/j.jksuci.2019.06.012. URL: https://linkinghub.elsevier.com/retrieve/pii/S1319157819304379.

[104] N. Kostikis et al. “Smartphone-based evaluation of parkinsonian hand tremor: Quantitative measurements vs clinical assessment scores”. In: *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, Aug. 2014, pp. 906–909. ISBN: 978-1-4244-7929-0. DOI: 10.1109/EMBC.2014.6943738. URL: http://ieeexplore.ieee.org/document/6943738/.

[105] Ikram Sumaiya Thaseen and Cherukuri Aswani Kumar. “Intrusion detection model using fusion of chi-square feature selection and multi class SVM”. In: *Journal of King Saud University - Computer and Information Sciences* 29.4 (Oct. 2017), pp. 462–472. ISSN: 13191578.

DOI: 10.1016/j.jksuci.2015.12.004. URL: https://linkinghub.elsevier.com/retrieve/pii/S1319157816300076.

[106] Daniel Berrar. "Cross-Validation". In: *Encyclopedia of Bioinformatics and Computational Biology*. Vol. 1-3. Elsevier, Mar. 2019, pp. 542–545. ISBN: 9780128114322. DOI: 10.1016/B978-0-12-809633-8.20349-X. URL: https://linkinghub.elsevier.com/retrieve/pii/B978012809633820349X%20http://arxiv.org/abs/1703.03167.

[107] Amalia Luque et al. "The impact of class imbalance in classification performance metrics based on the binary confusion matrix". In: *Pattern Recognition* 91 (July 2019), pp. 216–231. ISSN: 00313203. DOI: 10.1016/j.patcog.2019.02.023.

[108] Jill Marjama-Lyons and William Koller. "Tremor-predominant Parkinson's disease. Approaches to treatment". In: *Drugs and Aging* 16.4 (Aug. 2000), pp. 273–278. ISSN: 1170229X. DOI: 10.2165/00002512-200016040-00003/METRICS. URL: https://link.springer.com/article/10.2165/00002512-200016040-00003.

[109] Andrew P Duker and Alberto J Espay. "Surgical Treatment of Parkinson Disease". In: *Neurologic Clinics* 31.3 (Aug. 2013), pp. 799–808. ISSN: 07338619. DOI: 10.1016/j.ncl.2013.03.007. URL: https://linkinghub.elsevier.com/retrieve/pii/S0733861913000224.

[110] Xinyi Wang et al. "Hand tremor detection in videos with cluttered background using neural network based approaches". In: *Health Information Science and Systems* 9.1 (Dec. 2021), p. 30. ISSN: 2047-2501. DOI: 10.1007/s13755-021-00159-3. URL: https://link.springer.com/10.1007/s13755-021-00159-3.

[111] Luis Sigcha et al. "Automatic Resting Tremor Assessment in Parkinsons Disease Using Smartwatches and Multitask Convolutional Neural Networks". In: *Sensors* 21.1 (Jan. 2021), p. 291. ISSN: 1424-8220. DOI: 10.3390/s21010291. URL: https://www.mdpi.com/1424-8220/21/1/291.

[112] E. Lohmann et al. "A multidisciplinary study of patients with early-onset PD with and without parkin mutations". In: *Neurology* 72.2 (Jan. 2009), pp. 110–116. ISSN: 0028-3878. DOI: 10.1212/01.wnl.0000327098.86861.d4. URL: https://www.neurology.org/lookup/doi/10.1212/01.wnl.0000327098.86861.d4.

[113] Mehrbakhsh Nilashi et al. "Predicting Parkinsons Disease Progression: Evaluation of Ensemble Methods in Machine Learning". In: *Journal of Healthcare Engineering* 2022 (Feb. 2022). Ed. by M. Praveen Kumar Reddy, pp. 1–17. ISSN: 2040-2309. DOI: 10.1155/2022/2793361. URL: https://www.hindawi.com/journals/jhe/2022/2793361/.

[114] Aliénor Vienne et al. “Inertial Sensors to Assess Gait Quality in Patients with Neurological Disorders: A Systematic Review of Technical and Analytical Challenges”. In: *Frontiers in Psychology* 8.MAY (May 2017), p. 262425. ISSN: 1664-1078. DOI: 10.3389/fpsyg.2017.00817. URL: http://journal.frontiersin.org/article/10.3389/fpsyg.2017.00817/full.

[115] Marco Sica et al. “Continuous home monitoring of Parkinsons disease using inertial sensors: A systematic review”. In: *PLOS ONE* 16.2 (Feb. 2021). Ed. by Fabio A. Barbieri, e0246528. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0246528. URL: https://dx.plos.org/10.1371/journal.pone.0246528.

# Appendix A

# Research activities schedule

In this appendix, the main tasks of this thesis, which were planned for a period of 36 months, are described in detail. Among the work packages, various deliverables or milestones were evaluated regarding their impact and research excellence.

**Task 1.** **(1st to 5th and 12th to 14th month.)** 1) Define, develop, and implement in MATLAB the algorithms to preprocess the raw data of angular velocity and linear acceleration from the available dataset and then extract specific kinematic features. 2) Replicate the algorithms developed in the programming language to be used on the web server.

**Task 2.** **(4th to 6th and 24th to 25th month)** 1) Evaluate the measured kinematic features using statistical analysis to select those features that demonstrate a high discriminatory power between the classification groups. 2) Repeat the same process for the expanded database.

**Task 3.** **(5th a 10th, 15th to 16th and 26th to 27th month)** 1) Develop classification models based on the selected features and then evaluate and select the best set of models. 2) Implement the selected models in the programming language that will be used on the web server. 3) Repeat the previous steps using the features identified as having high discriminatory power in the extended database.

**Task 4.** **(11th to 18th month)** Develop a preliminary version of the mobile application on Android or iOS, and the webserver to record and store data from new subjects and test the performance of the classification models developed previously.

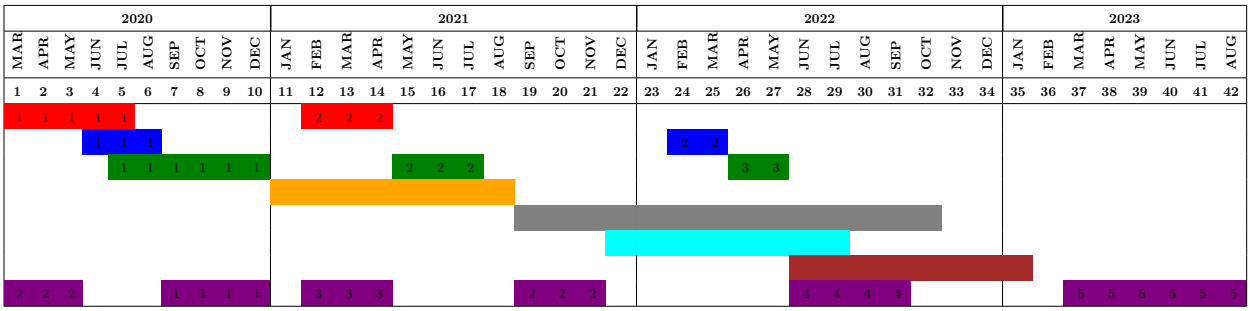
**Task 5.** **(19th to 32nd month)** Plan and execute, with the cooperation of specialists from the Department of Movement Disorders in different hospitals, the registration of new patients with Parkinson's disease (confirmed by SPECT), patients with essential tremor (confirmed

by a clinical evaluation), and healthy subjects.

**Task 6.** **(22nd to 29th month)** Analyze the expanded database to determine the influence of different operational parameters (sampling rate, weight of the recording device, dataset size, sensor axes, and sensor type) on the development of classification models that will be used in the final application.

**Task 7.** **(28th to 35th month)** Develop the final version of the mobile application and the web server and validate their performance with a small group of confirmed patients from collaborating healthcare institutions.

**Deliverables/milestones** 1) Write the research proposal to be evaluated as a candidate for a Ph.D. degree. 2) Write an article for a relevant journal with a high impact factor in Biomedical Engineering, Software, or Movement Disorders. 3) Prepare presentations of the results to communicate them to the scientific community, research centers, and international conferences. 4) Promote the use of this future open-access application and its web server in hospitals to validate its performance and expand the database of patients with tremor disorders. 5) Write the doctoral thesis for evaluation.



# Appendix B

# Conference Paper: *Using Machine Learning and Accelerometry Data for Differential Diagnosis of Parkinsons Disease and Essential Tremor*

# Using Machine Learning and Accelerometry Data for Differential Diagnosis of Parkinson's Disease and Essential Tremor

Julián D. Loaiza Duque[1,2], Andrés M. González-Vargas[1,2(✉)], Antonio J. Sánchez Egea[3(✉)], and Hermán A. González Rojas[3]

[1] Universidad Autónoma de Occidente, Santiago de Cali, Colombia
amgonzalezv@uao.edu.co
[2] Grupo de Investigación en Ingeniería Biomédica G-BIO, Santiago de Cali, Colombia
[3] Universidad Politécnica de Cataluña, Barcelona, Spain
antonio.egea@upc.edu

**Abstract.** Parkinson's disease (PD) and Essential Tremor (ET) are the most common tremor syndromes in the world. Currently, a specific Single Photon Emission Computed Tomography ($^{123}$I-FP-CIT SPECT) has proven to be an effective tool for the diagnosis of these diseases (97% sensitivity and 100% specificity). However, this test is invasive and expensive, and not all countries can have a SPECT system for an accurate differential diagnosis of PD patients. Clinical evaluation by a neurologist remains the gold standard for PD diagnosis, although the accuracy of this protocol depends on the experience and expertise of the physician. Wearable devices have been found to be a potential tool to help in differential diagnosis of PD and ET in early or complex cases. In this paper, we analyze the linear acceleration of the hand tremor recorded with a built-in accelerometer of a mobile phone, with a sampling frequency of 100 Hz. This hand tremor signal was thoroughly analyzed to extract different kinematic features in the frequency domain. These features were used to explore different Machine Learning methods to automatically classify and differentiate between healthy subjects and hand tremor patients (HETR Group) and, subsequently, patients with PD and ET (ETPD Group). Sensitivity of 90.0% and Specificity of 100.0% were obtained with classifiers of the HETR group. On the other hand, classifiers with Sensitivity ranges from 90.0% to 100.0% and Specificity from 80% to 100% were obtained for the ETPD group. These results indicate that the method proposed can be a potential tool to help the clinicians on differential diagnosis in complex or early hand tremor cases.

**Keywords:** Parkinson's Disease · Essential Tremor · Machine Learning · Accelerometry · Wearable device


J. C. Figueroa-García et al. (Eds.): WEA 2019, CCIS 1052, pp. 368–378, 2019.
https://doi.org/10.1007/978-3-030-31019-6_32

## 1 Introduction

Tremor is an involuntary, rhythmic and oscillatory movement of a part of the body [1]. It is not seen during sleep and its effects are commonly observed in the fingers, hands, legs, head and voice [2]. The limbs and head, when not supported, show a slight tremor called physiological tremor, which is generally of low amplitude and interferes only with fine motor control [1,3]. Physiological tremor is usually not visible or symptomatic, unless it is increased by fatigue or anxiety, whereas pathological tremor is usually visible and persistent [1].

Parkinson's disease (PD) and Essential Tremor (ET) are the most common tremor syndromes worldwide [4,5]. The differentiation between PD and ET can sometimes be difficult at early stages or patients without a family history of PD, and misdiagnosis rates can reach up to 25%, even when they are handled by a specialist in movement disorders [4,7–9]. Typically, PD is characterized by resting tremors and ET by postural or kinetic tremors [5]. However, some PD patients may have postural tremor [5] and some ET patients may have resting tremors during disease progression [10,11]. Accordingly, an early and accurate diagnosis is fundamental for treatment selection [4,5,10]. Early treatment of PD reduces or prevents disability and the need for support to maintain the quality of life, whereas incorrectly prescribing PD medication to ET patients is ineffective and exposes them to potential and serious side effects [5]. Nowadays, dopamine transporter (DAT) imaging using Single Photon Emission Computed Tomography (SPECT) with appropriate tracers ($^{123}$I-FP-CIT) has proven to be an efficient tool for diagnosing PD [4,5,12]. This technique is a high cost test and its use is limited to a few developed countries worldwide. Additionally, it is an invasive test with a radioactive fluid that requires patient compatibility, which may present limitations for its use.

Wearable devices are currently being widely investigated in the movement disorder field, because they can help clinicians in the differentiation between PD and ET. Several works have been published on this topic. Uchida et al. [11] used a triaxial accelerometer to measure the intensity and frequency of hand tremor in resting, posture, writing and walking conditions in subjects with ET and PD patients. They stated that tremor is attenuated during walking in ET subjects with resting tremor and increased in PD patients. Recently, Bernhard et al. [13] studied the gait and balance deficit by using wearables at the lower back and the ankle. They denoted that wearable devices let us assess the progression of movement disorders and response to treatment. Additionally, Wile et al. [14] made a classification of patients with PD and ET via calculation and analysis of the Mean Harmonic Power by using a smartwatch accelerometer. Thanks to that, they noted that compared to an analog accelerometer, a smartwatch device can provide accurate and relevant information for differential diagnosis between PD and ET subjects, based on postural tremor. Locatelli et al. [5] recorded hand tremors during resting, postural and kinetic tasks using a wearable sensor to differentiate PD and ET patients. They observed that, in the frequency domain, the execution of resting tasks showed a predominance of PD over ET tremors, while the data provided by postural and kinetic tasks stand out in

ET subjects. On the other hand, some researchers have used Discrete Wavelet Transforms and Support Vector Machines to differentiate between the two hand tremor conditions. Woods et al. [3] developed an offline application that uses a mobile phone accelerometer to perform the diagnosis and classification of PD and ET patients. Also, Surangsrirat et al. [9] classified PD and TE patients based on temporal angular velocity fluctuations, recorded with a 6-DOF inertial measurement unit. Additionally, Kramer et al. [15] recorded electromyography (EMG) and accelerometry (ACC) signals to distinguish between different types of tremor through Wavelet Coherence Analysis (WCA). They stated that WCA is superior to a standard coherence analysis and could be a useful additional tool for discriminating between tremor types, when the result obtained with other methods is inconclusive. Furthermore, Nanda et al. [8] used the Wavelet transform to extract EMG and ACC signal features. These features combined with an Artificial Neural Network were used to perform a quantitative classification of ET and PD. Finally, Raza et al. [16] compared the diagnosis obtained by using wearable devices with respect to the early diagnosis made by a specialist. In this work, machine learning methods were used to perform the differential classification between PD patients and patients with other movement disorders.

In previous works [4,6], methods for the differential diagnosis of subjects with movement disorders using the built-in accelerometer of a mobile phone were proposed. The proposed method in [4] allows to characterize and recognize the discriminatory features of hand tremor in patients with PD and ET. The present work uses the same data to implement several machine learning algorithms and kinematic indexes that could enhance the discrimination features and, ultimately, improve the sensibility and specificity not only between PD and ET, but also other types of tremor. We expect this method will be extremely useful to aid physicians in the differential diagnosis of complex or early cases.

## 2 Materials and Methods

Figure 1 shows the different steps of the method we use. These include signal recording using a smartphone, signal processing classification methods using Matlab.

The demographic characteristics of the subjects, the method for recording and preprocessing of the acceleration data used in this study is described in [4]. Data preprocessing, kinetic feature extraction, training and validation of classifiers were carried out using Matlab v. R2017b (Mathworks Inc., USA). Figure 2 summarizes all the tasks performed for the data structuring. Recorded data were initially preprocessed to remove noise components associated with respiration, pulse, or any sudden high-frequency movement. In addition, Power Spectral Density was calculated for each acceleration signal, from which six kinetic features were extracted. Finally, the set of kinetic features of each subject was properly labeled to structure the data.

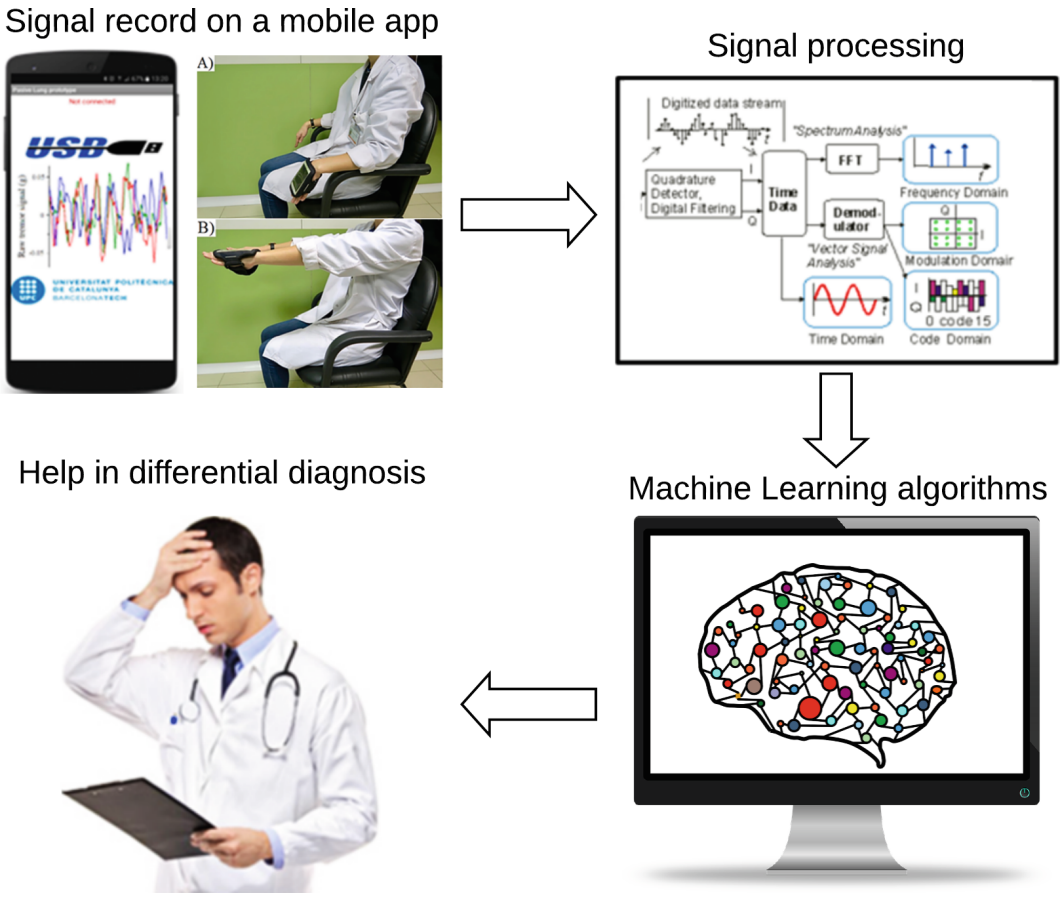



**Fig. 1.** Schematic of differential diagnosis system for PD and ET subjects.

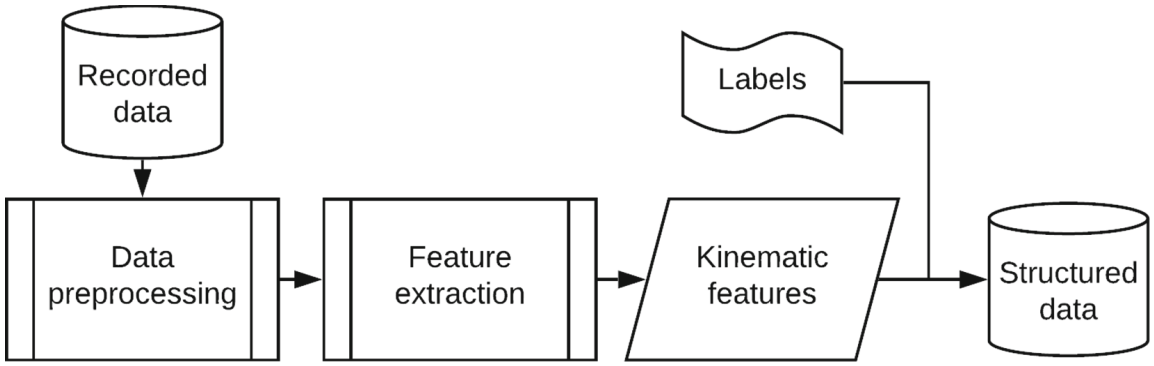



**Fig. 2.** Data structuring process: recording, preprocessing, featuring and labeling.

### 2.1 Subjects

A total of 52 subjects (17 patients with PD, 16 patients with ET, 12 healthy subjects and 7 patients with inconclusive diagnosis) were recorded in the Movement Disorders Unit of the Hospital Clínic of Barcelona between October 2015 and December 2016 [4]. All the patients had visually evidences of hand tremor and were diagnosed or had strong indications of PD or ET. Patients had scores of 1 or 2 on the Fahn-Tolosa-Marín scale for ET and Unified Parkinson's Disease

Rating Scale (UPDRS) for PD patients, a SPECT test confirmed all the patients with PD.

### 2.2 Data

The data was recorded with the triaxial accelerometer of an iPhone 5S using the SensorLog application software [17], and sent to a computer for further analysis. The subjects were seated in an armrest chair and the smartphone was placed on the dorsum of the most affected hand in patients or in the dominant hand in healthy subjects. Records of 30 s with a sampling frequency of 100 Hz were taken. Additionally, two arm positions were studied: (1) Rest position (Position A): the subject rests his forearm on the upper part of the armrest and (2) Arms stretched (Position B): the subject keeps both upper limbs fully extended.

### 2.3 Preprocessing

In early PD, the full triad of symptoms and clinical signs (resting tremor, bradykinesia and rigidity) may not be fully manifested [18,19] and usually the first indication for PD is resting tremor with moderate amplitude and low frequency (4–6 Hz), however, some PD patients may also present postural tremor with a medium frequency of 6 to 8 Hz [5]. ET is characterized by posture or kinetic tremors with a medium frequency (5–8 Hz) [5], although some patients may have tremors at rest during disease progression [10,11]. Besides the preprocessing described in the aforementioned study, the signals were filtered using a 10th order Butterworth filter with cut-off frequencies of 1 and 16 Hz [20], in which the PD and ET frequencies are found. Breathing, pulse or any sudden high frequency movement during recordings were also removed with this filter. Since the analysis was performed in the frequency domain, Power spectral density was calculated using Welch's periodogram by averaging 3s segments of signal recording with 50% overlap of Hanning's windows. The average power spectral densities of the linear accelerations were calculated to find the kinematic indices that allow us to differentiate hand tremor differences.

### 2.4 Feature Extraction

Figure 3 exhibits a Normalized Power Spectral Density (PSD) of tremor of an ET subject. It also illustrates the kinetic features calculated from the spectral power analysis: Median Power Frequency (MPF), Power Dispersion (PD), Peak Power Frequency (PPF), Harmonic Index (HI), Relative Power Contribution to the first harmonic (RPC) and Relative Energy (RE) to compare PD and ET subjects.

- **MPF:** Frequency at the power distribution center.
- **PD:** Frequency band, centered on MPF, which contains 90% of the total power.
- **PPF:** Frequency at which the maximum power is found.

– **HI:** Quotient between the area under the power spectral density curve and a rectangle bounded on the sides by the frequency band of interest (0–20 Hz) and vertically from 0 to PPF.
– **RPC:** Quotient between the power spectral density of harmonics found between a frequency division threshold ($f_{th}$) and 20 Hz and the total normalized power spectral density between 0 and 20 Hz.
– **RE:** Quotient between the normalized power spectral densities of resting ($PSD_r$) and posture ($PSD_p$) in the frequency range 0 to 20 Hz.

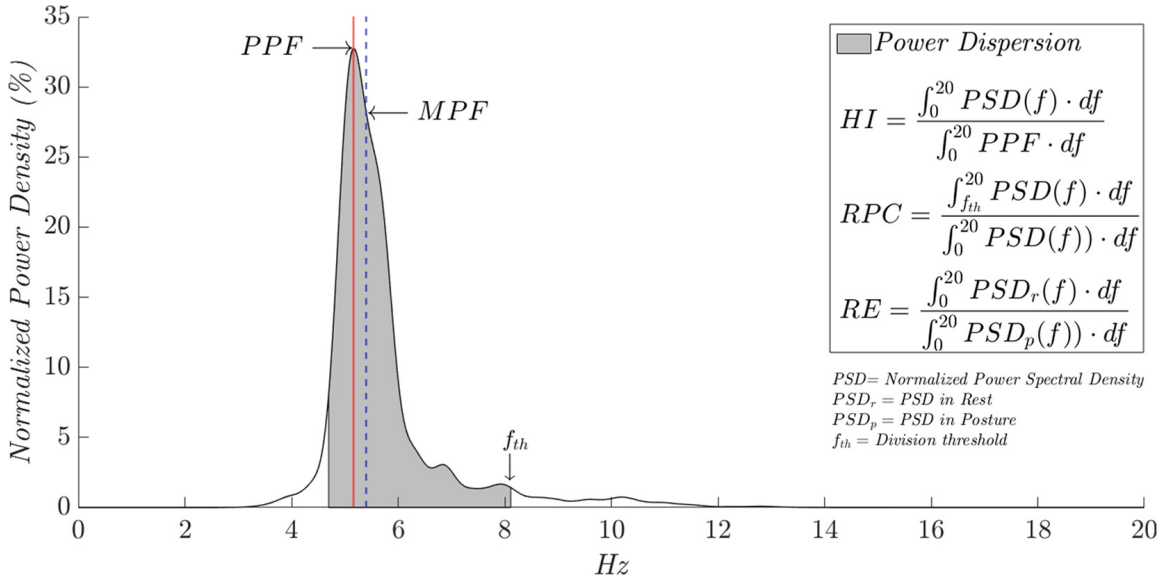



**Fig. 3.** A normalized spectral power density of tremor in an ET subject

In particular, RE and RPC features were added to enhance the differentiation between PD and ET [4], since their tremor frequency components are different in resting or posture conditions. Theoretically, PD patients should present a higher total spectral power of resting tremor than postural tremor, and in the opposite way for ET patients. The set of features extracted from the data of each subject were respectively labeled according to two classification groups:

1. **HETR** Group:

– Tremor patients - **TR** (Positive Class)
– Healthy subjects - **HE** (Negative Class)

2. **ETPD** Group:

– Parkinson's Disease - **PD** (Positive Class)
– Essential Tremor - **ET** (Negative Class)

Therefore, it is possible to classify subjects between HE and TR and, within subjects identified as TR differentiate between PD and ET patients.

### 2.5 Training, Validation and Selection of Classification Models

Figure 4 shows all the tasks performed in the process of training, validation and selection of classification models. All data were randomly divided into two sets (training set and validation set) with a proportion of 70-30. For the training set, a total of 63 combinations of features and classification methods were tested with Machine Learning algorithms with and without Principal Component Analysis (PCA), 23 with PCA + 23 without PCA.

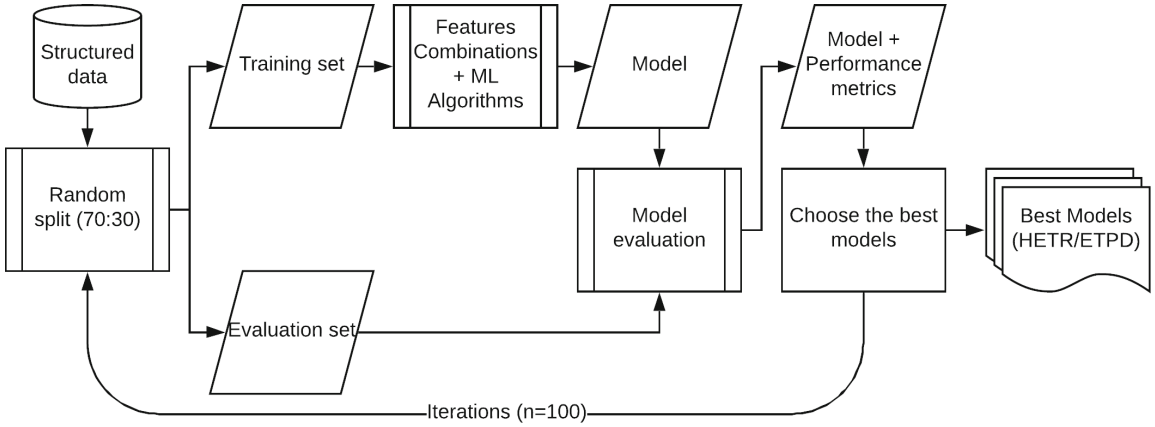



**Fig. 4.** Training, validation and selection of Classification Models

The performance of the classification models was evaluated by using a 6-fold cross validation. For each classification model, the accuracy and Area Under the Curve (AUC) for non-parametric receiver operating characteristic were estimated from the classification probabilities resulting of cross validation. Afterwards, validation sets were used to calculate Sensitivity (Eq. 1), Specificity (Eq. 2) and SSMean (Eq. 3), which is the average value of specificity and sensibility. In this context, Sensitivity defines the ability of a classification model to detect a positive case, that is, to detect patients with tremor in the group of HETR or patients with PD in the group of ETPD. Furthermore, Specificity defines the ability of the classification model to identify negative cases, being healthy subjects in the group of HETR Group or patients with ET in the group of ETPD.

$$Sensitivity = \frac{TP}{TP + FN} \tag{1}$$

$$Specificity = \frac{TN}{TN + FP} \tag{2}$$

$$SSMean = \frac{Sensitivity + Specificity}{2} \tag{3}$$

Validation and training processes were iterated 100 times for the same feature combinations and classification methods. This ensured that the training was carried out with a varied set of data, so that classification models with different

performance levels were considered. After all iterations, Sensitivity, Specificity and SSMean were calculated for each classification model. Classification models with the highest average values of SSMean in the groups of HETR and ETPD were identified. The best 10 classification models after these 100 iterations process were listed and studied in the results section.

## 3 Results

The output results after applying the iteration methodology is listed in Table 1. All the results reported here are obtained by testing on validation sets. The 10 best classification models come from a total of 2898 classification models, due to the feature and classification methods combinations. The average column represents the mean behavior during 100 iterations (in which training and validation data were randomized), whereas the best case column shows the best performance among all iterations.

**Table 1.** Healthy vs. Trembling subjects discrimination. Top 10 Classification Models with the highest SSMean values in the HETR group. **PCA:** Principal Components Analysis, **Sen:** Sensitivity **and Spe:** Specificity

| Features | Method | PCA | Averages | | | Best Case | | |
|---|---|---|---|---|---|---|---|---|
| | | | Sen | Spe | SSMean | Sen | Spe | SSMean |
| PPF + MPF | Quadratic Discriminant | No | 71.5 | 99.4 | 85.4 | 90.0 | 100.0 | 95.0 |
| MPF + HI | Quadratic Discriminant | No | 71.3 | 99.5 | 85.4 | 90.0 | 100.0 | 95.0 |
| PPF + MPF + PD | Quadratic Discriminant | No | 71,2 | 99.0 | 85.1 | 90.0 | 100.0 | 95.0 |
| MPF | Quadratic Discriminant | No | 70.7 | 99.4 | 85.0 | 90.0 | 100.0 | 95.0 |
| MPF | Quadratic Discriminant | Yes | 70.7 | 99.4 | 85.0 | 90.0 | 100.0 | 95.0 |
| PPF + MPF + HI | Quadratic Discriminant | No | 73.2 | 96.9 | 85.0 | 90.0 | 100.0 | 95.0 |
| PPF + MPF | Quadratic Discriminant | Yes | 70.2 | 99.8 | 85.0 | 90.0 | 100.0 | 95.0 |
| PPF + MPF + PD | Quadratic Discriminant | Yes | 70,2 | 99.8 | 85.0 | 90.0 | 100.0 | 95.0 |
| MPF + PD | Quadratic Discriminant | Yes | 70.2 | 99.4 | 84.8 | 90.0 | 100.0 | 95.0 |
| MPF + PD + HI | Quadratic Discriminant | No | 70.5 | 98.9 | 84.7 | 90.0 | 100.0 | 95.0 |

It is noticeable that all the models used the Quadratic Discrimination method, and the common kinematic feature for all of them was MPF. This suggests that the MPF feature may provide a significant differentiation between HE and TR. The classification model with the highest SSMean average was obtained using both MPF and PPF features. In other words, compared to the other models, this classification model had the best performance in most of the 100 iterations. The models that presented the maximum SSMean values are considered on the top of the table, because of the combined good results in sensibility and specificity. Note that for all classification models the best cases had an SSMean value of 95.0% (90.0% Sensitivity and 100.0% Specificity). Moreover, Table 2 shows the best 10 classification models with the highest SSMean values

**Table 2.** Parkinson's Disease vs. Essential Tremor patient discrimination. Top 10 Classification Models with the highest SSMean values in the ETPD group. **PCA:** Principal Components Analysis, **Sen:** Sensitivity **and Spe:** Specificity

| Features | Method | PCA | Averages | | | Best Case | | |
|---|---|---|---|---|---|---|---|---|
| | | | Sen | Spe | SSMean | Sen | Spe | SSMean |
| RPC + RE + HI | Logistic Regression | No | 69.2 | 85.4 | 77.3 | 100.0 | 90.0 | 95.0 |
| RPC + RE + MPF + HI | Logistic Regression | No | 69.5 | 83.0 | 76.3 | 100.0 | 90.0 | 95.0 |
| RPC + RE + PPF + HI | Logistic Regression | No | 67.7 | 84.5 | 76.1 | 100.0 | 80.0 | 90.0 |
| RPC + RE | Logistic Regression | No | 66.7 | 84.8 | 75.8 | 100.0 | 100.0 | 100.0 |
| RPC + RE + PPF | Logistic Regression | No | 66.4 | 84.9 | 75.7 | 90.0 | 90.0 | 90.0 |
| RE + PPF + HI | Logistic Regression | No | 66.6 | 83.9 | 75.3 | 100.0 | 100.0 | 100.0 |
| RE | Medium KNN | No | 73.8 | 76.4 | 75.1 | 100.0 | 100.0 | 100.0 |
| RE | Cubic KNN | No | 73.8 | 76.4 | 75.1 | 100.0 | 100.0 | 100.0 |
| RE | Medium KNN | Yes | 73,8 | 76.4 | 75.1 | 100.0 | 100.0 | 100.0 |
| RE | Cubic KNN | Yes | 73.8 | 76.4 | 75.1 | 100.0 | 100.0 | 100.0 |

for the group of ETPD, where it can be noted which method and kinematic are essential to distinguish between PD and ET subjects.

The first six classification models used the Linear Regression method, whereas the last four used different types of KNN algorithms. Note that the kinematic feature RE is used in all the classification models to differentiate between the two groups. This is consistent with the results obtained in [4], since with this feature a significant differentiation was obtained between patients with PD and ET (84.4% discrimination accuracy). RPC is a feature that also had significant performance in the previous paper. Classification models that use these two features are on the top five. The present work found that the best case of the classification model that combines the Logistic Regression method and these two kinematic features (RPC and RE) obtained a SSMean value of 100.0% (100.0% Sensitivity and 100.0% Specificity). Table 2 shows five other cases in which an SSMean value of 100.0% was obtained. These are promising results to develop a helpful tool for clinicians for the differential diagnosis of PD and ET. However, a larger database will be needed in order to further validate these results.

## 4 Conclusion

The potential benefits of using Machine Learning for classification of patients with hand tremor was investigated in this paper. The main findings drawn from this research are, firstly, that the linear acceleration is able to provide significant information for an appropriate classification of healthy subjects and patients with tremor and, ultimately, differentiate between PD and ET subjects. The effectiveness of such differentiation depends substantially on the correct selection and evaluation of the classifier to be implemented. Secondly, during the training of the classifiers, it was possible to identify outstanding performance of kinetic features combinations and classification methods. In particular,

Quadratic Discriminant method combined with MPF feature were the most relevant combination to differentiate healthy from pathological subjects, whereas Logistic Regression method combined with RE and RPC features were crucial to differentiated PD from ET subjects.

As future work, the methodology presented in this paper will be implemented to analyze the angular velocity signal of the gyroscope built-in the mobile device. In this way, it will be possible to determine if the angular velocity assess a higher performance level than that obtained with the linear acceleration analysis. Finally, a low-cost app will be developed to provide relevant information to clinicians to help in clinical evaluation of the patients with hand tremor in the first stages.

**Acknowledgements.** This work was supported by Dirección de Investigaciones y Desarrollo Tecnológico (DIDT) of Universidad Autónoma de Occidente, Project 19INTER-308: "Herramienta no invasiva de bajo costo para el diagnóstico diferencial temprano en pacientes con Parkinson y Temblor Esencial" and by the Serra Húnter program (Generalitat de Catalunya) reference number UPC-LE-304.

## References

1. Bhatia, K.-P., et al.: Consensus statement on the classification of tremors. From the task force on tremor of the International Parkinson and Movement Disorder Society. Mov. Disord. **33**, 75–87 (2018). https://doi.org/10.1002/mds.27121
2. Bhavana, C., Gopal, J., Raghavendra, P., Vanitha, K.-M., Talasila, V.: Techniques of measurement for Parkinson's tremor highlighting advantages of embedded IMU over EMG. In: 2016 International Conference on Recent Trends in Information Technology (ICRTIT), pp. 1–5. IEEE (2016)
3. Woods, A.-M., Nowostawski, M., Franz, E.-A., Purvis, M.: Parkinson's disease and essential tremor classification on mobile device. Pervasive Mob. Comput. **13**, 1–12 (2014). https://doi.org/10.1016/j.pmcj.2013.10.002
4. Barrantes, S., et al.: Differential diagnosis between Parkinson's disease and essential tremor using the smartphone's accelerometer. PLoS ONE **12**, e0183843 (2017). https://doi.org/10.1371/journal.pone.0183843
5. Locatelli, P., Alimonti, D.: Differentiating essential tremor and Parkinson's disease using a wearable sensor – a pilot study. In: 2017 7th IEEE International Workshop on Advances in Sensors and Interfaces (IWASI), pp. 213–218. IEEE (2017)
6. González Rojas, H.-A., Cuevas, P.-C., Zayas Figueras, E.-E., Foix, S.-C., Sánchez Egea, A.-J.: Time measurement characterization of stand-to-sit and sit-to-stand transitions by using a smartphone. Med. Biol. Eng. Comput. **56**, 879–888 (2018). https://doi.org/10.1007/s11517-017-1728-5
7. Miller, D.-B., O'Callaghan, J.-P.: Biomarkers of Parkinson's disease: present and future. Metabolism **64**, S40–S46 (2015). https://doi.org/10.1016/j.metabol.2014.10.030
8. Nanda, S.K., Lin, W.-Y., Lee, M.-Y., Chen, R.-S.: A quantitative classification of essential and Parkinson's tremor using wavelet transform and artificial neural network on sEMG and accelerometer signals. In: 2015 IEEE 12th International Conference on Networking, Sensing and Control, pp. 399–404. IEEE (2015)

9. Surangsrirat, D., Thanawattano, C., Pongthornseri, R., Dumnin, S., Anan, C., Bhidayasiri, R.: Support vector machine classification of Parkinson's disease and essential tremor subjects based on temporal fluctuation. In: 2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pp. 6389–6392. IEEE (2016)
10. Papengut, F., Raethjen, J., Binder, A., Deuschl, G.: Rest tremor suppression may separate essential from Parkinsonian rest tremor. Parkinsonism Relat. Disord. **19**, 693–697 (2013). https://doi.org/10.1016/j.parkreldis.2013.03.013
11. Uchida, K., Hirayama, M., Yamashita, F., Hori, N., Nakamura, T., Sobue, G.: Tremor is attenuated during walking in essential tremor with resting tremor but not Parkinsonian tremor. J. Clin. Neurosci. **18**, 1224–1228 (2011). https://doi.org/10.1016/j.jocn.2010.12.053
12. Algarni, M., Fasano, A.: The overlap between essential tremor and Parkinson disease. Parkinsonism Relat. Disord. **46**, S101–S104 (2018). https://doi.org/10.1016/j.parkreldis.2017.07.006
13. Bernhard, F.-P., et al.: Wearables for gait and balance assessment in the neurological ward - study design and first results of a prospective cross-sectional feasibility study with 384 inpatients. BMC Neurol. **18**, 114 (2018). https://doi.org/10.1186/s12883-018-1111-7
14. Wile, D.-J., Ranawaya, R., Kiss, Z.-H.-T.: Smart watch accelerometry for analysis and diagnosis of tremor. J. Neurosci. Methods **230**, 1–4 (2014). https://doi.org/10.1016/j.jneumeth.2014.04.021
15. Kramer, G., Van der Stouwe, A.-M.-M., Maurits, N.-M., Tijssen, M.-A.-J., Elting, J.-W.-J.: Wavelet coherence analysis: a new approach to distinguish organic and functional tremor types. Clin. Neurophysiol. **129**, 13–20 (2018). https://doi.org/10.1016/j.clinph.2017.10.002
16. Raza, M.-A., Chaudry, Q., Zaidi, S.-M.-T., Khan, M.-B.: Clinical decision support system for Parkinson's disease and related movement disorders. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1108–1112. IEEE (2017)
17. Sensorlog (Version 1.9.4) Mobile application software. http://itunes.apple.com. Accessed 24 Apr 2019
18. Brooks, D.-J.: Parkinson's disease: diagnosis. Parkinsonism Relat. Disord. **18**, S31–S33 (2012). https://doi.org/10.1016/S1353-8020(11)70012-8
19. Arvind, R., Karthik, B., Sriraam, N., Kannan, J.-K.: Automated detection of PD resting tremor using PSD with recurrent neural network classifier. In: 2010 International Conference on Advances in Recent Technologies in Communication and Computing, pp. 414–417. IEEE (2010)
20. Jeon, H., et al.: Automatic classification of tremor severity in Parkinson's disease using a wearable device. Sensors **17**, 2067 (2017). https://doi.org/10.3390/s17092067

# Appendix C

# Journal Paper: *Angular Velocity Analysis Boosted by Machine Learning for Helping in the Differential Diagnosis of Parkinsons Disease and Essential Tremor*

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON BODY AREA NETWORKS

Received April 24, 2020, accepted May 6, 2020, date of publication May 11, 2020, date of current version May 22, 2020.

*Digital Object Identifier 10.1109/ACCESS.2020.2993647*

# Angular Velocity Analysis Boosted by Machine Learning for Helping in the Differential Diagnosis of Parkinson's Disease and Essential Tremor

**JULIÁN D. LOAIZA DUQUE[1,2,3], ANTONIO J. SÁNCHEZ EGEA[3], THERESA REEB[4], HERNÁN A. GONZÁLEZ ROJAS[3], AND ANDRÉS M. GONZÁLEZ-VARGAS[1,2]**

[1]Department of Automation and Electronics, Universidad Autónoma de Occidente (UAO), Cali 760030, Colombia
[2]Research Group in Biomedical Engineering (G-BIO), Universidad Autónoma de Occidente (UAO), Cali 760030, Colombia
[3]Department of Mechanical Engineering, Universitat Politècnica de Catalunya (UPC), 08034 Barcelona, Spain
[4]Department of Mechanical Engineering, Ostbayerische Technische Hochschule Amberg–Weiden, 92224 Amberg, Germany

Corresponding author: Antonio J. Sánchez Egea (antonio.egea@upc.edu)

This work was supported in part by the Dirección de Investigaciones y Desarrollo Tecnológico (DIDT) of Universidad Autónoma de Occidente, through Herramienta no invasiva de bajo costo para el diagnóstico diferencial temprano en pacientes con Parkinson y Temblor Esencial, under Project 19INTER-308, and in part by the Serra Húnter Programme, Generalitat de Catalunya, in 2018, under Grant UPC-LE-304.

**ABSTRACT** Recent research has shown that smartphones/smartwatches have a high potential to help physicians to identify and differentiate between different movement disorders. This work aims to develop Machine Learning models to improve the differential diagnosis between patients with Parkinson's Disease and Essential Tremor. For this purpose, we use a mobile phone's built-in gyroscope to record the angular velocity signals of two different arm positions during the patient's follow-up, more precisely, in rest and posture positions. To develop and to find the best classification models, diverse factors were considered, such as the frequency range, the training and testing divisions, the kinematic features, and the classification method. We performed a two-stage kinematic analysis, first to differentiate between healthy and trembling subjects and then between patients with Parkinson's Disease and Essential Tremor. The models developed reached an average accuracy of 97.2 $\pm$ 3.7% (98.5% Sensitivity, 93.3% Specificity) to differentiate between Healthy and Trembling subjects and an average accuracy of 77.8 $\pm$ 9.9% (75.7% Sensitivity, 80.0% Specificity) to discriminate between Parkinson's Disease and Essential Tremor patients. Therefore, we conclude, that the angular velocity signal can be used to develop Machine Learning models for the differential diagnosis of Parkinson's disease and Essential Tremor.

**INDEX TERMS** Differential diagnosis, Parkinson's disease, essential tremor, gyroscope, kinematic analysis, machine learning.

## I. INTRODUCTION

Tremor is a compulsory and oscillatory movement of a part of the body [1]. Its effects are primarily visible in the limbs, head, and voice [2]. Physiological tremor is usually of low amplitude and interferes only with fine motor control. In most cases, it is not visible or symptomatic, except when increased by fatigue or anxiety [1], [3]. On the contrary, pathological tremor is usually visible and constant [1]. Parkinson's disease (PD) and Essential Tremor (ET) are the most common tremor syndromes worldwide [4], [5]. Distinguishing between PD and ET can be difficult in the early stages of the diseases or for patients without a family history of PD. The risk of incorrect diagnosis is high; even specialists in movement disorders may have a rate of up to 25% false positives or negatives [4], [6]–[8]. Typically, resting tremors are associated with PD, whereas postural or kinetic tremors associate with ET [5]. However, some PD patients may develop postural tremor [5], and some ET patients may develop resting tremors during the progression of the disease [9], [10]. Early diagnosis is fundamental to ensure adequate treatment of the patient and to prevent harmful side-effects [4], [5], [9]. Nowadays, dopamine transporter (DAT) imaging using Single Photon Emission Computed Tomography (SPECT) with appropriate tracers ($^{123}$I-FP-CIT) is the most reliable technique for diagnosing PD [4], [5], [11]. However, the test

The associate editor coordinating the review of this manuscript and approving it for publication was Diep N. Nguyen.

is costly and therefore limited to economically developed countries. Additionally, it is an invasive test with a radioactive fluid that requires patient compatibility, which may limit its applicability.

Therefore, it is a current topic of research to develop fast and non-invasive techniques for the early and reliable diagnosis of PD. Unlike the kinematic position information captured with optical movement detection systems [12], the accelerometry analysis is currently a hot topic in the biomechanical field. It records the motion information of physical activity based on wearable devices [13]. In this sense, extensive research on the use of wearable devices in the field of movement disorders is underway, with numerous papers published on these topics. Uchida *et al.* [10] employed a triaxial accelerometer to measure the severity and frequency of hand tremors in patients with ET and PD under conditions of rest, posture, writing, and walking. They observed that resting tremor is attenuated during walking in patients with ET and increased in patients with PD. Recently, Bernhard *et al.* [14] studied the gait and balance deficit by using wearables fixed at the lower back and the ankle. They denoted that wearable gadgets could assess the progression of movement disorders and the response to the treatment of the disease. Wile *et al.* [15] classified patients with PD and ET via calculation and analysis of the Mean Harmonic Power using a smartwatch accelerometer. They noted that, compared to an analog accelerometer, a smartwatch device could provide accurate and relevant information for the differential diagnosis between PD and ET subjects. Locatelli and [5] recorded hand tremors during resting, postural, and kinematic tasks using a wearable sensor to differentiate PD and ET patients. They observed that, in the frequency domain, the execution of resting tasks showed a predominance of PD over ET tremors. In contrast, the data provided by postural and kinetic tasks stand out in ET subjects.

Some researchers have used Machine Learning (ML) to differentiate between the two tremor conditions. Woods *et al.* [3] developed an offline application that uses a mobile phone accelerometer to perform the diagnosis and classification of PD and ET patients. Surangsrirat *et al.* [9] classified PD and ET patients based on temporal angular velocity fluctuations, recorded with a 6-DOF inertial measurement unit. Kramer *et al.* [16] combined Electromyography (EMG), and Accelerometry (ACC) signals to distinguish between different types of tremor through Wavelet Coherence Analysis (WCA). They stated that WCA is superior to a standard coherence analysis and could be a useful additional tool for discriminating between tremor types when the result obtained with other methods is inconclusive. Nanda *et al.* [7] used the Wavelet transform to extract EMG and ACC signal features. These features, combined with an Artificial Neural Network, were used to perform a quantitative classification of ET and PD. Finally, Raza *et al.* [17] compared the diagnosis obtained by using wearable devices with the early diagnosis made by a specialist. They also used ML methods to perform the differential classification between PD and other movement disorders. Besides, in previous works, we proposed different methods for the differential diagnosis of the two diseases using the mobile phone's built-in triaxial accelerometer [4], [18], [19]. The developed methods allow to characterize and recognize the discriminative features of hand tremor in PD and ET patients and to use ML algorithms to improve the differentiation between them.

This work aims to use the same methodology to evaluate the angular velocity data, recorded with the mobile phone's built-in gyroscope, and to build ML models to differentiate healthy subjects (HS) and tremor patients (TP) and, subsequently, within the subjects identified as TP to discriminate PD patients from ET patients. These models are performed based on two different frequency ranges and three group divisions. We expect this method to be an additional tool to help the physician in case of uncertainty and undecided diagnosis of the diseases.

## II. MATERIALS AND METHODS

Fig. 1 illustrates the different steps that compose the methodology developed in this work: Signal recording with a mobile phone, data analysis, and model training and testing. The demographic characteristics of the subjects, the method of recording, and the preprocessing of the dataset are described in Barrantes *et al.* [4]. The whole process was carried out in Matlab v. R2019b (MathWorks Inc., USA) on a computer with an Intel i5-9600K processor at 3.70 GHz, 16 GB of RAM and an NVIDIA GeForce GTX 1650 graphics card with 4 GB of V-RAM.

### A. PATIENTS AND DATASET DESCRIPTION

The dataset used in this study includes recordings of 19 PD patients, 20 ET patients, and 12 HS from the Movement Disorders Unit of the Hospital Clinic of Barcelona between October 2015 and December 2016 [4]. All the patients had visual evidence of hand tremors and were diagnosed with strong indications of PD or ET. Patients had scores of 1 or 2 on the Fahn-Tolosa-Marín scale for ET and the Unified Parkinson's Disease Rating Scale (UPDRS) for PD patients. A SPECT test confirmed all the patients with PD.

The angular velocity signals were collected with the built-in triaxial gyroscope of an iPhone 5S using SensorLog application [20]. The smartphone was placed on the dorsum of the most affected hand in TP or the dominant hand in HS while sitting in an armrest chair. Tremor signals were recorded with a frequency of 100 Hz and an average duration of $35.66 \pm 4.08$ s, $35.42 \pm 3.42$ s, and $33.30 \pm 3.27$ s for HS, ET, and PD subjects, respectively. As shown in Fig. 1, two-arm positions were studied: 1) Rest (Position A), the subject rests his forearm on the upper part of the armrest, and 2) Posture (Position B), the subject keeps both upper limbs fully extended.

### B. DATA ANALYSIS

One of the clinical signs and symptoms of PD is tremor at rest with moderate amplitudes and low frequencies from 4 to

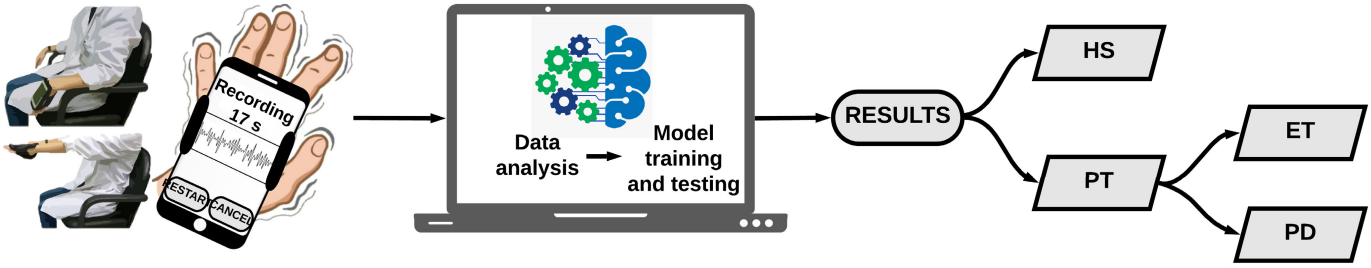



**FIGURE 1.** Schematic of the methodology for the differential diagnosis of PD and ET patients.

6 Hz [9], [21]. In contrast, ET is characterized by postural or kinetic tremors with mean frequency values between of 5 to 8 Hz [15], [22]. Furthermore, physiological tremor is in the frequency band of 8 to 12 Hz [23]. Based on this, the dataset is preprocessed as follows in order to extract the kinematic features: artifacts generated by starting and ending the signal recording were eliminated by cutting approximately 2 seconds on both sides of the signals. Two 10th order Butterworth filters with cut-off frequencies of 3 to 10 Hz [11] and 1 to 16 Hz [24], where PD and ET are found, were implemented separately in order to identify an optimal frequency range for feature extraction. Additionally, these filters allow reducing the sensor offsets and drifts due to various physical phenomena such as motion artifacts [17], [25]. Figure 2 shows the time-domain signal of PD, ET, and HS subjects in posture position before and after signal processing.

Since the analysis was performed in the frequency domain, Power Spectral Density (PSD) was calculated. For each of the three spatial directions, a Welch's periodogram averaging segments of the signal recording of 3s with a 50% overlap of Hanning's window was applied. The PSD average of the angular velocity components was calculated and normalized. The resulting average was used to calculate kinematic indexes that allow the identification and classification of subjects with pathological tremor and differentiate them between PD and ET. The kinematic features are briefly explained below:

- **Median Power Frequency (MPF):** Frequency at which the PSD is halved.
- **Power Bandwidth (PB):** Frequency band, centered around the MPF, which contains 90% of the total power.
- **Peak Power Frequency (PPF):** Frequency at which the maximum power is located.
- **Harmonic Index (HI):** Quotient between the area under the PSD curve and a rectangle bounded on the sides by the frequency band of interest ($f_l$ -$f_h$) and the Peak Power (PP).

$$HI = \frac{\int_{f_{th}}^{f_h} PSD(f) \cdot df}{PP \cdot (f_h - f_l)} \tag{1}$$

- **Relative Power Contribution to the first harmonic (RPC):** Quotient between the PSD of harmonics found between a frequency division threshold ($f_{th}$) and $f_h$ and the PSD between $f_l$ and $f_h$.

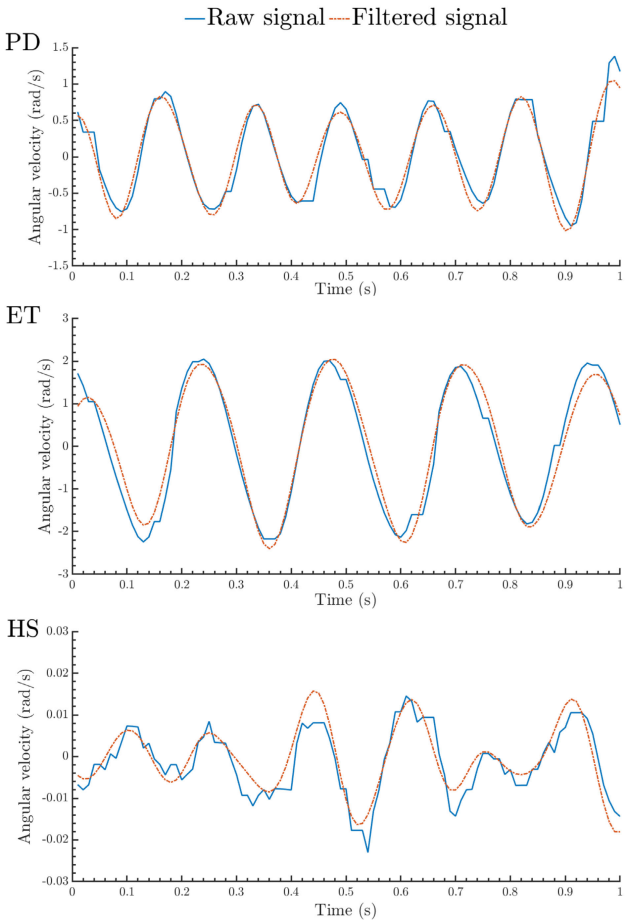



**FIGURE 2.** Time-domain signal of PD, ET and HS subjects in posture position before and after signal processing.

$$RPC = \frac{\int_{f_{th}}^{f_h} PSD(f) \cdot df}{\int_{f_l}^{f_h} PSD(f) \cdot df} \tag{2}$$

- **Relative Energy (RE):** Quotient between the normalized PSD of resting ($PSD_A$) and posture ($PSD_B$) in the frequency range of $f_l$ to $f_h$.

$$RE = \frac{\int_{f_l}^{f_h} PSD_A \cdot df}{\int_{f_l}^{f_h} PSD_B \cdot df} \tag{3}$$

- **Harmonic Index Ratio (HIR):** Quotient between the harmonic indexes of resting and posture position.

$$HIR = \frac{HI_A}{HI_B} \tag{4}$$

- **Sum of Maximum Power (SMP):** Sum of the power value at the PP of resting and posture position.

$$SMP = PP_A + PP_B \tag{5}$$

After extracting the feature matrix of the subjects, they were labeled as follows:

1) **Case 1: TP vs. HS**
   - **TP** (Tremor patients) - Positive Class.
   - **HS** (Healthy subjects) - Negative Class.
2) **Case 2: PD vs. ET**
   - **PD** (Parkinson's Disease) - Positive Class.
   - **ET** (Essential Tremor) - Negative Class.

Since thirteen features have been extracted per subject, we used feature selection algorithms [26] to reduce the dimensionality of the resulting matrix and to select a subset of a maximum of five features to create the classification models. This allows to reduce the training time of the models and to focus on the features that provide the highest differentiation between both Cases' classes. We used the Chi-square test and the Unbiased Tree method to estimate, separately, the importance of each feature [27], [28]. For each test, the five features with the highest importance values were identified. The features that matched in both tests were chosen for further analysis. This process was carried out in two frequency ranges: 1-16 Hz and 3-10 Hz.

### C. MODEL TRAINING AND TESTING

The classification models designed differ in four aspects:

1) **The frequency range of analysis.** As mentioned in the previous subsection, the kinematic features were extracted in two different frequency ranges (1-16 Hz and 3-10 Hz) to identify which range is optimal for differentiating between physiological and pathological tremors and, subsequently, between pathological tremors.
2) **The proportion of training and testing data.** For each of the cases presented, the dataset was randomly divided into three different proportions (30/70, 50/50, and 70/30), ensuring that both positive and negative classes were distributed at the same ratio in each training and testing set. Table 1 details, for both cases in all proportions, the class ratios obtained in the training and testing sets.
   The reason why we decided to use three different divisions and not one, as commonly implemented in ML, was to evaluate the influence of the data distribution to obtain high-performance models.
3) **The kinematic features used.** Using the features extracted and selected during the data analysis, we identified all the possible combinations of features that can be generated, from a single feature to the whole of them. Since we set 5 as the maximum number of features, for some cases, up to 31 combinations of features were obtained. These feature combinations allowed us to evaluate the discriminatory ability the features can reach individually or in combination using the classification methods that implement them.
4) **The classification method used to train the model.** The classification methods used for training the models were developed based on the Matlab *Classification Learner* app. This app offers a variety of supervised ML methods to classify data, including decision trees, discriminant analysis, Support Vector Machines, Logistic Regression, Nearest Neighbors, Naive Bayes, and ensemble classification. There are several default configurations of hyperparameters of these methods in the app, offering a total of 25 different configurations for the training of classification models. We integrated all configurations into a script and applied them to the dataset.

**TABLE 1.** Training and testing set class ratios.

| | Case 1: TP vs. HS | | | | Case 2: PD vs. ET | | | |
|---|---|---|---|---|---|---|---|---|
| Division | Training | | Testing | | Training | | Testing | |
| (%) | PC | NC | PC | NC | PC | NC | PC | NC |
| 30 / 70 | 12 | 4 | 27 | 8 | 6 | 6 | 13 | 14 |
| 50 / 50 | 20 | 6 | 19 | 6 | 10 | 10 | 9 | 10 |
| 70 / 30 | 27 | 8 | 12 | 4 | 13 | 14 | 6 | 6 |

PC, Positive class. NC, Negative class.

Given the number of combinations of features that were possible to obtain and the diverse configurations of the classification methods, we obtained 775 different classification models for some cases. After setting the training sets, the testing sets were used to calculate Accuracy, Sensitivity and Specificity. We defined Sensitivity as the capacity of a classification model to identify positive cases, that is, to identify TP in Case 1 or PD subjects in Case 2. On the contrary, Specificity is defined as the ability of the classification model to identify negative cases, being HS in Case 1 or ET subjects in Case 2. All training and testing processes were randomly iterated 100 times for the same combinations of features and classification methods in each of the three training/testing divisions. Consequently, a different level of performance was obtained in each iteration for each model. After all iterations, the average values of Accuracy, Sensitivity, and Specificity obtained for each classification model were calculated. The three best classification models for Cases 1 and 2 were identified based on the output classification metrics. Fig. 3 summarizes the whole process that was implemented for the development and selection of the classification models.

## III. RESULTS

We divide the results of this work into two subsections. In the first part, we evaluate the model's capacity to differentiate TP from HS. In the second part, we analyze the model's ability to differentiate patients with PD and ET.

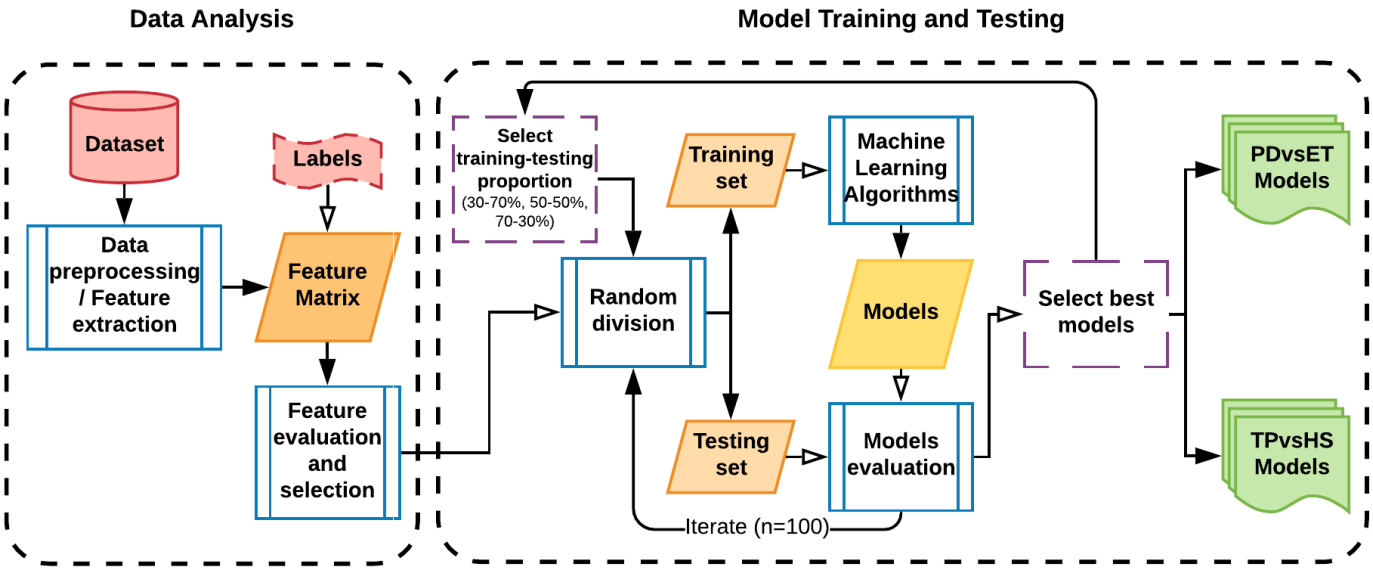



**FIGURE 3.** Process diagram for the development and selection of classification models.

**TABLE 2.** Evaluation and selection of kinematic features for the differentiation of tremor and healthy subjects.

| Feature | Position | 3 - 10 Hz | | 1 - 16 Hz | |
|---|---|---|---|---|---|
| | | CS | UT | CS | UT |
| MPF | A | 3.23 | 0.04 | 3.02 | 0.07 |
| | B | 3.36 | 0.02 | 3.98 | 0.03 |
| PB | A | 3.98 | 0.03 | 3.98 | 0.06 |
| | B | **8.12** | **0.07** | **11.74** | **0.14** |
| PPF | A | 3.49 | 0.01 | 3.73 | 0.06 |
| | B | 4.76 | 0.03 | 8.12 | 0.08 |
| HI | A | **4.76** | **0.06** | 4.76 | 0.07 |
| | B | **9.01** | **0.08** | **9.01** | **0.14** |
| RPC | A | 3.42 | 0.01 | 4.96 | 0.10 |
| | B | **6.97** | **0.07** | **10.82** | **0.15** |
| RE | A/B | 1.87 | 0.00 | 3.98 | 0.00 |
| HIR | A/B | 0.78 | 0.01 | 0.35 | 0.00 |
| SMP | A+B | **11.74** | **0.12** | **10.82** | **0.16** |

A, rest position. B, postural position. CS, Chi-square test. UT, Unbiased Tree method. Bolded values correspond to the five features with the highest discriminative values in both tests.

### A. DIFFERENTIATION OF TREMOR PATIENTS AND HEALTHY SUBJECTS

Table 2 shows the results of the evaluation and selection of features for distinguishing between TP and HS. In the 3 to 10 Hz frequency analysis, the five features with the highest values were identical in both tests. These features were: SMP, $RPC_B$, $HI_B$, $HI_A$, and $PB_B$. In the 1-16 Hz frequency analysis, four of the five features identified by both tests coincided: SMP, $RPC_B$, $HI_B$, and $PB_B$.

The upper and left side of Figure 4 shows the best models for the differentiation of TP and HS in the frequency range of 3-10 Hz, sorted by the three training/testing divisions. For each division, the top 3 models were identified and listed based on their average metrics. The SMP feature is present in all nine models, while $PB_B$, $HI_B$, and $RPC_B$ are present in two of them. The best performing classification model shows an average accuracy of $94.3 \pm 5.6\%$ (95.9% sensitivity, 89.5% specificity), and an average computational cost of $6.7 \pm 0.7$ ms. This model was achieved in a 70/30 division, using the SMP feature and the Linear SVM method. Although there are a variety of classification methods among the nine listed, in both the 30/70 and 50/50 divisions, the best model implemented the Logistic Regression method and the SMP feature. On the right side, the figure visualizes the best models obtained in the frequency analysis from 1 to 16 Hz in all training/testing divisions. Again, the three best models were selected based on their average performances. All models in this frequency range use SMP as a discriminatory feature, while the $PB_B$ feature is applied in eight of them. The best model shows an average accuracy of $97.2 \pm 3.7\%$ (98.5% sensitivity, 93.3% specificity), and an average computational cost of $105.8 \pm 1.9$ ms. There is only one model that implements a single feature, SMP, using a 70/30 division and the Medium Tree method. The rest of the models implement Ensemble Subspace KNN method and combine various features. Note that the average computational cost of the models that use the Medium Tree method with a single feature is considerably smaller than those obtained with the models that use the Ensemble Subspace KNN method and multiple features.

### B. DIFFERENTIATION OF PARKINSON'S DISEASE PATIENTS VS. ESSENTIAL TREMOR PATIENTS

Table 3 shows the evaluation and selection of features for the differentiation of PD and ET patients. In the 3-10 Hz frequency analysis, the five features identified in each test, separately, were the same: SMP, HIR, RE, $RPC_A$, and $MPF_A$. In the frequency range of 1-16 Hz, only three of the five features coincided: HIR, RE, and $RPC_A$.

The bottom left side of Figure 4 depicts the best models for the differentiation of PD and ET in the frequency range

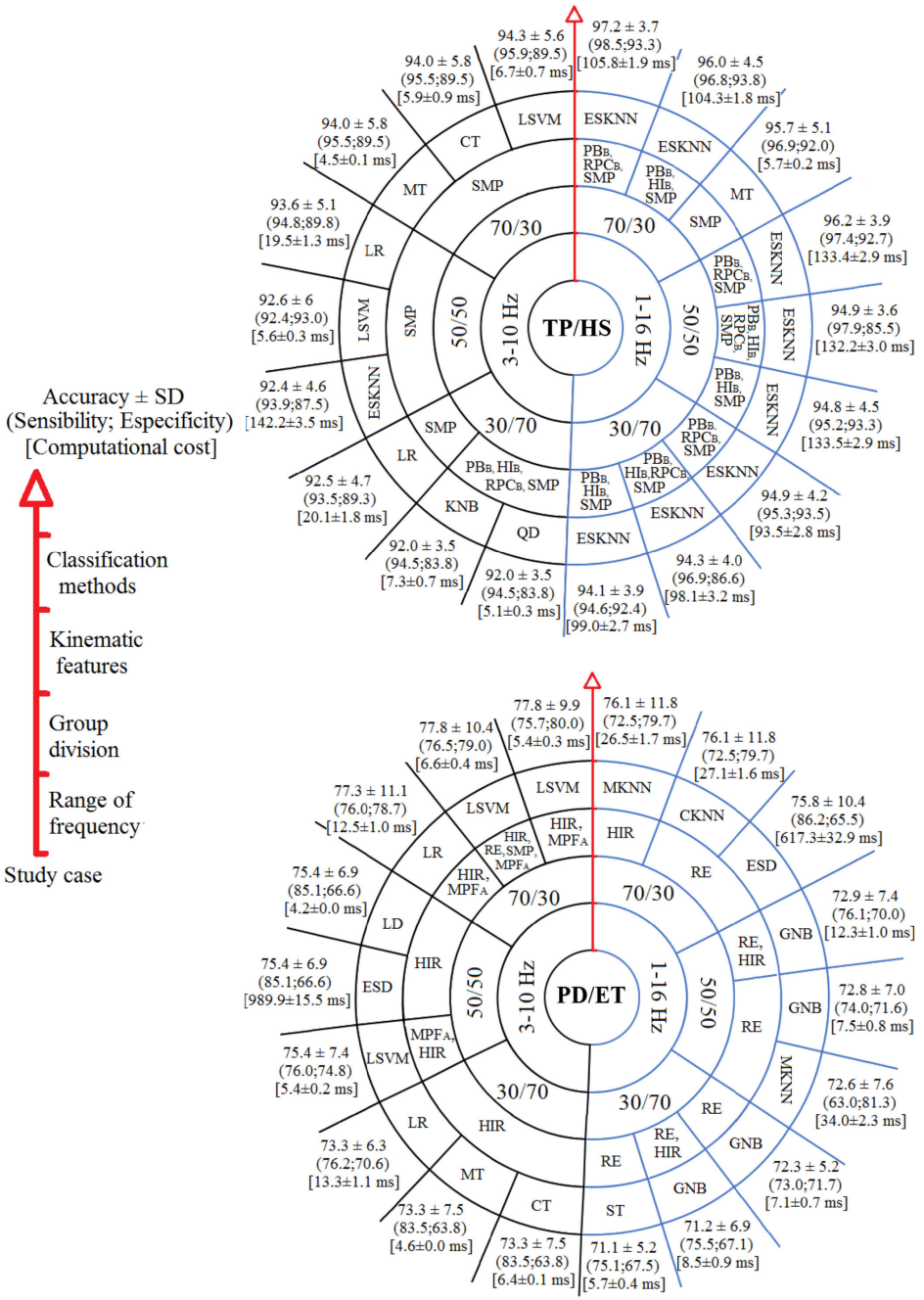



**FIGURE 4.** **Output results of the machine-learning algorithm based on study case, range of frequency, kinematic features, and classification methods.**

of 3-10 Hz. The top 3 models in each training/testing division are listed, sorted by their average performance values. The HIR feature seems to provide significant information for the differentiation of tremor patients, since it is present in all the models depicted. The best overall performance was achieved in the 70/30 division, combining the HIR and $MPF_A$ features and using the Linear SVM method. This model showed an average accuracy of 77.8 ± 9.9% (75.7% sensitivity, 80.0% specificity), and an average computational cost of 5.4 ± 0.3 ms. The right side of the figure visualizes the models with the best performances for the differentiation of PD and ET in the frequency range from 1 to 16 Hz. Again, the best model can be found in the 70/30 division, with an average accuracy of 76.1 ± 11.8% (72.5% sensitivity, 79.7% specificity) and an average computational cost of 26.5 ± 1.7 ms. The feature that is present in most of the models is RE, being used in eight of the nine models shown. In the 30/70 and 50/50 divisions, the two best classification models use the Gaussian Naive Bayes method. In contrast, in the 70/30 division, the two best performances were obtained with two different configurations of the KNN method, obtaining the same average accuracy.

**TABLE 3. Evaluation and selection of kinematic features for the differentiation of tremor subjects: PD vs. ET.**

| Feature | Position | 3 - 10 Hz | | 1 - 16 Hz | |
|---|---|---|---|---|---|
| | | CS | UT | CS | UT |
| MPF | A | **2.70** | **0.03** | 0.54 | 0.02 |
| | B | 0.13 | 0.00 | 0.12 | 0.00 |
| PB | A | 0.62 | 0.03 | 1.12 | 0.05 |
| | B | 1.12 | 0.02 | 0.62 | 0.01 |
| PPF | A | 0.62 | 0.02 | 0.62 | 0.01 |
| | B | 0.05 | 0.01 | 0.01 | 0.00 |
| HI | A | 0.62 | 0.02 | 0.62 | 0.04 |
| | B | 1.63 | 0.03 | 0.34 | 0.02 |
| RPC | A | **1.91** | **0.04** | **1.37** | **0.05** |
| | B | 0.12 | 0.01 | 1.63 | 0.02 |
| RE | A/B | **3.79** | **0.07** | **5.20** | **0.09** |
| HIR | A/B | **3.34** | **0.07** | **2.10** | **0.06** |
| SMP | A+B | **1.91** | **0.04** | 1.91 | 0.01 |

A, rest position. B, postural position. CS, Chi-square test. UT, Unbiased Tree method. Bolded values correspond to the five features with the highest discriminative values in both tests.

## IV. DISCUSSION

The results obtained in this work show that the characterization and differentiation between tremor in PD and ET are possible with a mobile phone's built-in gyroscope. The accuracy of the tremor differentiation using this sensor is comparable to the performance obtained using a mobile phone's built-in accelerometer [4], [19]. Although there is a clear difference between the number of TP (39 in total) and HS (12 in total), the accuracy of the models differentiating the two conditions is high. This is due to the differences in the frequency components of the tremors that characterize both classes. By analyzing the entire data in the frequency domain, we were able to highlight these differences. Since the PSD in HS can be up to 1000 times lower than in trembling subjects, we obtained higher accuracy values than in [17], (82.43%), even though their dataset was considerably larger than ours. Other studies [27], [29] reported accuracy values of 82% to 100%; however, their groups of trembling subjects only included PD patients. In [8], [30], wearable sensors (accelerometers and gyroscopes) were used to extract features that allowed the implementation of ML algorithms for the differentiation between PD and ET, reaching accuracies of 96% to 100%. In [8], the analysis was performed in the time domain and kinetic tremors instead of tremors in posture were analyzed. The study performed in [30] uses accelerometry data, registers each patient for a recording time of five minutes, and uses a newly introduced posture as well as statistical analysis of the data's frequency components to differentiate the subjects. Compared to those studies, our classification models were developed to be used during clinical follow-up, where simple postures and short recording times are required. The accuracy values reaches in our study are lower than those in [8], [30], for two reasons. Firstly, they both registered more subjects which improves the predictive ability of the models. Secondly, the accuracy values we represent in this study are average values of 100 random iterations in three training/testing divisions. In single iterations, the classification models developed for PD/ET differentiation reached similar values. Moreover, since the aim of this work was to evaluate whether the angular velocity signal could help to differentiate tremor subjects using ML, we considered the use of the default configurations of the ML methods to be enough. In future works, we intend to analyze in detail how to adjust the hyperparameters of the implemented models to optimize their discriminative capacity.

The frequency ranges used to develop the models generated significant differences regarding their performance. For the differentiation of TP and HS, the average accuracy values obtained in the frequency analysis from 1 to 16 Hz are higher than those obtained in the analysis from 3 to 10 Hz. These differences could exist because the frequency range from 3 to 10 Hz includes only a part of the area in which physiological tremors occur (8 to 12 Hz) [23], whereas the analysis of 1 to 16 Hz includes its full range. Nevertheless, the models generated in the 1 to 16 Hz range require complicated methods and more kinematic features. For the differentiation of PD and ET patients, the models analyzed in the 3-10 Hz frequency range show better performance compared to those in the 1-16 Hz frequency range. These performance differences could be directly related to the dominant frequencies of the two tremor types. As mentioned in the Data Analysis subsection, both PD and ET tremors are located in a frequency range between 4 and 8 Hz [9], [15], [21], [22]. Thus, the extraction of kinematic features within a frequency range of 3 to 10 Hz eliminates unwanted effects that are introduced by frequencies outside the area of interest.

It is noticeable that the variability in the performance of the PD/ET models listed is relatively high (5.2% to 11.8%). This variability is influenced by the presence of atypical patient data in each iteration since, as mentioned previously, there are PD patients who experience postural tremors [5] and ET patients who show tremors at rest during disease progression [9], [10]. Other variability factors are the training/testing divisions, as the data distribution influences the performance of the classification models. As expected, the classification models show better performances the higher the percentage of data in the training set. Analyzing Figure 4, the models for differentiating TP and HS exhibit a difference of 3.1% when comparing 30/70 and 70/30 divisions combined with identical features (SMP, $RPC_B$, and $PB_B$) and the same classification method (Ensemble Subspace KNN). The models for differentiating PD and ET show a difference of 4.0% when comparing 30/70 and 70/30 divisions combined with the same features (HIR) and classification method (Logistic regression).

Based on the presumption that the frequency components of the pathological tremor are higher in either of the two positions studied, SMP and HIR were introduced to improve the differentiation between the tremor types. RE and RPC features were proposed in [4] to improve the differentiation between PD and ET patients, as their tremor frequency components are different under resting or postural conditions. Theoretically, PD patients should have higher amplitudes of tremor at rest (position A) than postural tremor (position B),

and vice versa for patients with ET. The results obtained in this work supported the above, the most significant feature for the differentiation of patients with PD and ET seems to be the novel HIR feature, as it was implemented in 12 of the 18 best models depicted in Figure 4. Also, as already observed in previous works [4], [19], RE and RPC features provide essential information. The RPC feature also contains relevant information for the differentiation of TP and HS in both analyzed frequency ranges. However, the SMP feature introduced in this study was most discriminative in several of the best models; high accuracy values were reached by only using this relative feature. Analyzing the implemented features, it is noticeable that some of them provide more accurate information for the differentiation of the subject according to the Case. The features extracted in the posture position were predominant in the models that differentiate between subjects in Case 1. In Case 2, there is a higher presence of features extracted in the resting position, which is consistent with the works of [5], [8].

As it was the intention to develop high-performance classifiers and avoid classification errors, only patients with a confirmed diagnosis of PD or ET were used to implement the ML models. However, this also means that the patients were already on treatment when they were registered, so their tremors intensity was remarkably low. For this reason, we consider that additional records should be performed on early-stage tremor patients to prevent the effects of medication [31] or surgical suppression [32], as these are possible causes of misclassification of patients. Another important topic regarding the development of high-performance models is the dataset size. Since the dataset for training and testing of the models was small, the ML models implemented in this study are limited in their performance. The dataset needs to be increased to develop highly accurate models. Therefore, in the second phase of the project, we aim to introduce a mobile application linked to a web server that allows adding new patient records to the already registered data. This phase will be realized through the collaboration of an international network of physicians and biomedical engineers using the application. By enlarging the dataset, we expect to improve the accuracy of the developed models or to create new models with even higher performance and lower computational cost.

## V. CONCLUSION

The angular velocity signal recorded by the gyroscope and boosted using ML algorithms has proven to be an effective method to differentiate between healthy subjects and tremor patients as well as between Parkinson's disease patients and Essential Tremor patients. This differentiation is substantially dependent on the correct selection and evaluation of classification methods and kinematic features, as well as on the processing and the size of the training data. The best model to differentiate HS and TP has an average accuracy of 97.2 $\pm$ 3.7% (98.5% Sensitivity, 93.3% Specificity). The average accuracy of the best model to differentiate tremor patients with PD and ET was 77.8 $\pm$ 9.9% (75.7% Sensitivity, 80.0% Specificity).

During the training of the models, we were able to identify outstanding performance for some combinations of kinematic features, such as SMP, $PB_B$, and $RPC_B$, for TP and HS differentiation, as well as HIR and $MPF_A$ for PD and ET differentiation. Regarding the classification methods, for the differentiation of TP and HS (Case 1), the best performances were reached with the Linear Support Vector Machine and Ensemble Subspace KNN methods. For the differentiation of PD and ET (Case 2), in the frequency analysis from 3 to 10 Hz, the best performance was also obtained with the Linear Support Vector Machine method. In contrast, in the 1-16 Hz range, the best performance was obtained with Medium K-nearest Neighbor method. In both cases, the Linear Support Vector Machine models present a lower computational cost compared to the KNN methods.

In future works, we want to combine the recordings of accelerometer and gyroscope sensor to obtain higher classification performances and reduce the training times. The optimized ML models developed in this research will be used to design a low-cost and non-invasive tool (mobile app) to support physicians in the differential diagnosis of the two diseases, particularly in developing countries where sophisticated diagnostic techniques such as $^{123}$I-FP-CIT SPECT are not available. Additionally, we expect that the use of this tool will help in patients with undecided diagnosis and, consequently, in choosing appropriate and opportune therapeutic actions.

## REFERENCES


[1] K. P. Bhatia, P. Bain, N. Bajaj, R. J. Elble, M. Hallett, E. D. Louis, J. Raethjen, M. Stamelou, C. M. Testa, and G. Deuschl, "Consensus statement on the classification of tremors. From the task force on tremor of the international parkinson and movement disorder society," *Movement Disorders*, vol. 33, no. 1, pp. 75–87, Jan. 2018, doi: 10.1002/mds.27121.

[2] C. Bhavana, J. Gopal, P. Raghavendra, K. M. Vanitha, and V. Talasila, "Techniques of measurement for Parkinson's tremor highlighting advantages of embedded IMU over EMG," in *Proc. Int. Conf. Recent Trends Inf. Technol. (ICRTIT)*, Apr. 2016, pp. 1–5, doi: 10.1109/ICRTIT.2016.7569560.

[3] A. M. Woods, M. Nowostawski, E. A. Franz, and M. Purvis, "Parkinson's disease and essential tremor classification on mobile device," *Pervas. Mobile Comput.*, vol. 13, pp. 1–12, Aug. 2014, doi: 10.1016/j.pmcj.2013.10.002.

[4] S. Barrantes, A. J. Sánchez Egea, H. A. González Rojas, M. J. Martí, Y. Compta, F. Valldeoriola, E. S. Mezquita, E. Tolosa, and J. Valls-Solè, "Differential diagnosis between Parkinson's disease and essential tremor using the smartphone's accelerometer," *PLoS ONE*, vol. 12, no. 8, Aug. 2017, Art. no. e0183843, doi: 10.1371/journal.pone.0183843.

[5] P. Locatelli and D. Alimonti, "Differentiating essential tremor and Parkinson's disease using a wearable sensor—A pilot study," in *Proc. 7th IEEE Int. Workshop Adv. Sensors Interfaces (IWASI)*, Jun. 2017, pp. 213–218, doi: 10.1109/IWASI.2017.7974254.

[6] D. B. Miller and J. P. O'Callaghan, "Biomarkers of Parkinson's disease: Present and future," *Metabolism*, vol. 64, no. 3, pp. S40–S46, Mar. 2015, doi: 10.1016/j.metabol.2014.10.030.

[7] S. K. Nanda, W.-Y. Lin, M.-Y. Lee, and R.-S. Chen, "A quantitative classification of essential and Parkinson's tremor using wavelet transform and artificial neural network on sEMG and accelerometer signals," in *Proc. IEEE 12th Int. Conf. Netw., Sens. Control*, Apr. 2015, pp. 399–404, doi: 10.1109/ICNSC.2015.7116070.

[8] D. Surangsrirat, C. Thanawattano, R. Pongthornseri, S. Dumnin, C. Anan, and R. Bhidayasiri, "Support vector machine classification of Parkinson's disease and essential tremor subjects based on temporal fluctuation," in *Proc. 38th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, Aug. 2016, pp. 6389–6392, doi: 10.1109/EMBC.2016.7592190.
[9] F. Papengut, J. Raethjen, A. Binder, and G. Deuschl, "Rest tremor suppression may separate essential from parkinsonian rest tremor," *Parkinsonism Rel. Disorders*, vol. 19, no. 7, pp. 693–697, Jul. 2013, doi: 10.1016/j.parkreldis.2013.03.013.
[10] K. Uchida, M. Hirayama, F. Yamashita, N. Hori, T. Nakamura, and G. Sobue, "Tremor is attenuated during walking in essential tremor with resting tremor but not parkinsonian tremor," *J. Clin. Neurosci.*, vol. 18, no. 9, pp. 1224–1228, Sep. 2011, doi: 10.1016/j.jocn.2010.12.053.
[11] M. Algarni and A. Fasano, "The overlap between essential tremor and parkinson disease," *Parkinsonism Rel. Disorders*, vol. 46, pp. S101–S104, Jan. 2018, doi: 10.1016/j.parkreldis.2017.07.006.
[12] E. Nikfekr, K. Kerr, S. Attfield, and E. D. Playford, "Trunk movement in Parkinson's disease during rising from seated position," *Movement Disorders*, vol. 17, no. 2, pp. 274–282, Mar. 2002, doi: 10.1002/mds.10073.
[13] G. Serrancolí, J. M. Font-Llagunes, and A. Barjau, "A weighted cost function to deal with the muscle force sharing problem in injured subjects: A single case study," *Proc. Inst. Mech. Eng., K, J. Multi-body Dyn.*, vol. 228, no. 3, pp. 241–251, Sep. 2014, doi: 10.1177/1464419314530110.
[14] F. P. Bernhard, J. Sartor, K. Bettecken, M. A. Hobert, C. Arnold, Y. G. Weber, S. Poli, N. G. Margraf, C. Schlenstedt, C. Hansen, and W. Maetzler, "Wearables for gait and balance assessment in the neurological ward–study design and first results of a prospective cross-sectional feasibility study with 384 inpatients," *BMC Neurol.*, vol. 18, no. 1, p. 114, Dec. 2018, doi: 10.1186/s12883-018-1111-7.
[15] D. J. Wile, R. Ranawaya, and Z. H. T. Kiss, "Smart watch accelerometry for analysis and diagnosis of tremor," *J. Neurosci. Methods*, vol. 230, pp. 1–4, Jun. 2014, doi: 10.1016/j.jneumeth.2014.04.021.
[16] G. Kramer, A. M. M. Van der Stouwe, N. M. Maurits, M. A. J. Tijssen, and J. W. J. Elting, "Wavelet coherence analysis: A new approach to distinguish organic and functional tremor types," *Clin. Neurophysiol.*, vol. 129, no. 1, pp. 13–20, Jan. 2018, doi: 10.1016/j.clinph.2017.10.002.
[17] M. A. Raza, Q. Chaudry, S. M. T. Zaidi, and M. B. Khan, "Clinical decision support system for Parkinson's disease and related movement disorders," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2017, pp. 1108–1112, doi: 10.1109/ICASSP.2017.7952328.
[18] H. A. González Rojas, P. C. Cuevas, E. E. Zayas Figueras, S. C. Foix, and A. J. Sánchez Egea, "Time measurement characterization of stand-to-sit and sit-to-stand transitions by using a smartphone," *Med. Biol. Eng. Comput.*, vol. 56, no. 5, pp. 879–888, May 2018, doi: 10.1007/s11517-017-1728-5.
[19] J. D. Loaiza Duque, A. M. González-Vargas, A. J. Sánchez Egea, and H. A. González Rojas, "Using machine learning and accelerometry data for differential diagnosis of parkinson's disease and essential tremor," in *Applied Computer Sciences in Engineering* (Communications in Computer and Information Science), vol. 1052. Springer, 2019, pp. 368–378.
[20] *Sensorlog (Version 1.9.4)* Mobile Application Software. Accessed: 2017. [Online]. Available: http://itunes.apple.com
[21] S. Padma G, S. Umesh, U. Asokan, and T. Srinivas, "Parkinsonian hand tremor measurement device based on fiber Bragg grating sensor," in *Proc. Int. Conf. Smart Sensors Syst. (IC-SSS)*, Dec. 2015, pp. 3–5, doi: 10.1109/SMARTSENS.2015.7873611.
[22] F. Hopfner and R. C. Helmich, "The etiology of essential tremor: Genes versus environment," *Parkinsonism Rel. Disorders*, vol. 46, pp. S92–S96, Jan. 2018, doi: 10.1016/j.parkreldis.2017.07.014.
[23] T. Novak and K. M. Newell, "Physiological tremor (8–12 Hz component) in isometric force control," *Neurosci. Lett.*, vol. 641, pp. 87–93, Feb. 2017, doi: 10.1016/j.neulet.2017.01.034.
[24] E. D. Louis, "Essential tremor then and now: How views of the most common tremor diathesis have changed over time," *Parkinsonism Rel. Disorders*, vol. 46, pp. S70–S74, Jan. 2018, doi: 10.1016/j.parkreldis.2017.07.010.
[25] O. Martinez Manzanera, J. W. Elting, J. H. van der Hoeven, and N. M. Maurits, "Tremor detection using parametric and non-parametric spectral estimation methods: A comparison with clinical assessment," *PLoS ONE*, vol. 11, no. 6, Jun. 2016, Art. no. e0156822, doi: 10.1371/journal.pone.0156822.
[26] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, Jan. 2003, doi: 10.1162/153244303322753616.
[27] N. Kostikis, D. Hristu-Varsakelis, M. Arnaoutoglou, and C. Kotsavasiloglou, "A smartphone-based tool for assessing parkinsonian hand tremor," *IEEE J. Biomed. Health Informat.*, vol. 19, no. 6, pp. 1835–1842, Nov. 2015, doi: 10.1109/JBHI.2015.2471093.
[28] I. S. Thaseen and C. A. Kumar, "Intrusion detection model using fusion of chi-square feature selection and multi class SVM," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 29, no. 4, pp. 462–472, Oct. 2017, doi: 10.1016/j.jksuci.2015.12.004.
[29] L. M. Gil, T. P. Nunes, F. H. S. Silva, A. C. D. Faria, and P. L. Melo, "Analysis of human tremor in patients with parkinson disease using entropy measures of signal complexity," in *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol.*, Aug. 2010, pp. 2786–2789, doi: 10.1109/IEMBS.2010.5626365.
[30] B. Zhang, F. Huang, J. Liu, and D. Zhang, "A novel posture for better differentiation between Parkinson's tremor and essential tremor," *Frontiers Neurosci.*, vol. 12, p. 317, May 2018, doi: 10.3389/fnins.2018.00317.
[31] J. Marjama-Lyons and W. Koller, "Tremor-predominant parkinson's disease," *Drugs Aging*, vol. 16, no. 4, pp. 273–278, Apr. 2000, doi: 10.2165/00002512-200016040-00003.
[32] A. P. Duker and A. J. Espay, "Surgical treatment of parkinson disease," *Neurologic Clinics*, vol. 31, no. 3, pp. 799–808, Aug. 2013, doi: 10.1016/j.ncl.2013.03.007.

**JULIÁN D. LOAIZA DUQUE** received the B.Eng. degree in biomedical engineering from the Universidad Autónoma de Occidente (UAO), Cali, Colombia, in 2017. He is currently pursuing the Ph.D. degree in mechanical, fluids, and aerospace engineering with the Universitat Politècnica de Catalunya. Since 2018, he has been a Research Assistant with the UAO. His research interests include biomedical signals and image processing, bio instrumentation, and artificial intelligence.

**ANTONIO J. SÁNCHEZ EGEA** received the M.Sc. degree in biomedical engineering from the Universitat de Barcelona, Barcelona, Spain, in 2011, and the Ph.D. degree in mechanical engineering from the Universitat Politècnica de Catalunya (UPC), Barcelona, in 2016. From 2016 to 2019, he held several postdoctoral positions at the Aeronautics Advanced Manufacturing Center, Bilbao, Spain, and the Pontificia Universidad Católica de Chile, Chile. He is currently an Assistant Professor with the UPC. His current research interests include biomechanical analysis, signal processing, machine learning, and advanced manufacturing processes and systems.

**THERESA REEB** received the B.Sc. degree in biomedical engineering from OTH Regensburg, Germany, in 2018. She is currently pursuing the M.Sc. degree in biomedical engineering with OTH Amberg–Weiden. In 2019, she started a research exchange with the Universidad Autónoma de Occidente and keeps working with them. Her current research interests are machine learning and signal analysis.

**HERNÁN A. GONZÁLEZ ROJAS** received the B.A. degree in mechanical engineering from the Universidad de Santiago de Chile, Chile, in 1993, and the Ph.D. degree in industrial engineering from the Universitat Politècnica de Catalunya (UPC), Barcelona, Spain, in 2009. He is an Associate Professor with the UPC and a member of the research group GAECE. His research interests include signal processing applied to biomedical applications, biomechanical analysis, biomedical instrumentation, and manufacturing processes.

**ANDRÉS M. GONZÁLEZ-VARGAS** received the B.Eng. degree in mechatronics engineering from the Universidad Autónoma de Occidente (UAO), in 2004, the M.Sc. degree in biomedical engineering from the Universitat de Barcelona, Barcelona, Spain, in 2011, and the Ph.D. degree in computer science, electronics, and electrical engineering from the Università di Pavia, Italia, in 2015. He is currently an Assistant Professor with the UAO, Colombia. His research interests include biomedical engineering, computational biology, engineering education, and biomedical informatics.

# S1 Appendix. Matlab code for Data Analysis

```
clearvars;
close all;
clc;
warning('off', 'all');
```

## Data preproccesing

Load the dataset, define the struct (**eKF**) where the kinematic features will be stored and create the categorical array (**PT**) to label the subjects.

```
load dataset
zA = zeros(length(dataset), 1);
eKF = struct('MPF_A', zA, 'PB_A', zA, 'PPF_A', zA, 'HI_A', zA, 'RPC_A', zA, 'MPF_B', ...
    zA, 'PB_B', zA, 'PPF_B', zA, 'HI_B', zA, 'RPC_B', zA, 'RE', zA, 'SMP', zA, 'HIR', ...
    zA, 'SUBJ', zA);
PT = categorical(zA);
```

Load the Butterworth filter (**btwFc3_10** or **btwFc1_16**), and set the number of samples to remove artifacts (**uS**).

```
load btwFc3_10.mat
% load btwFc1_16.mat
uS = 175;
```

Sampling frequency

```
sKF = 100;
```

Set up a FOR loop to preprocess and extract the kinematic features of each subject from the dataset. The odd rows are the resting records (*Position A*) and, the even rows are the posture records (*Position B*).

```
for i = 1:length(dataset)
```

Assign subject signals to **data**

```
    data = dataset{i, 1};
```

Cut approximately 2 seconds from both sides of the signals to eliminate artifacts generated by turning the recording on and off.

```
    data = data(uS:end-uS, :);
```

10th order Butterworth filter to reduce the sensor offsets and drifts due to various physical phenomena such as motion artifacts

```
    btwfData = filter(Hd, data(:, 7:9));
```

Welch filter with non-overlapping Hanning window of 3 seconds

```
    [pxx, ~] = pwelch(btwfData(:, 1), 300, [], 25000, sKF, 'psd');
    [pyy, ~] = pwelch(btwfData(:, 2), 300, [], 25000, sKF, 'psd');
    [pzz, f] = pwelch(btwfData(:, 3), 300, [], 25000, sKF, 'psd');
```

Set the borderline frequency (**fLow - fHigh**) for analysis.

```
    fLow = find(f == 3);
%     fLow = find(f==1);
    fHigh = find(f == 10);
%     fHigh = find(f==16);
```

Average the *Power Spectral Density* (*PSD*) of each axis and, then normalize (**nPSD**) the resulting average (**aPSD**).

```
    aPSD = (pxx + pyy + pzz) ./ 3;
    f = f(fLow:fHigh);
    aPSD = aPSD(fLow:fHigh);

    sPSD = sum(aPSD);
    nPSD = (aPSD ./ sPSD);
```

## Feature extraction

```
    if strcmp(dataset{i, 2}(5), 'A')
        pos = 'A';
        iter = '(i)';
    else
        pos = 'B';
        iter = '(i-1)';
    end
```

### Median Power Frequency (MPF) feature

```
    sumPD = sum(nPSD);
    posMedian = find(cumsum(nPSD/sumPD) <= sumPD/2, 1, 'last');
    eval(['eKF', iter, '.MPF_', pos, '= f(posMedian);']);
```

### Power Bandwidth (PB) feature

```
    lowPos = find(cumsum(nPSD(1:posMedian)/sumPD, 'reverse') ...
        >= 0.45, 1, 'last') + 1; %#ok<NASGU>
    highPos = posMedian + find(cumsum(nPSD(posMedian:end)./sumPD) ...
        < 0.4501, 1, 'last') - 1; %#ok<NASGU>
    eval(['eKF', iter, '.PB_', pos, '= f(highPos) - f(lowPos);']);
```

### Peak Power Frequency (PPF) feature

```
    [~, posPPF] = max(aPSD);
```

```
    eval(['eKF', iter, '.PPF_', pos, '=f(posPPF);']);
```

### Harmonic index (HI) feature

```
    rArea(1:length(aPSD)) = aPSD(posPPF); %#ok<NASGU>
    eval(['eKF', iter, '.HI_', pos, '=trapz(f,aPSD)./trapz(f,rArea);']);
```

### Relative power contribution to the first harmonic (RPC) feature

```
    eval(['eKF', iter, '.RPC_', pos, '=trapz(f(highPos:end),aPSD(highPos:end))', ...
        './trapz(f,aPSD);']);
```

### Relative Energy (RE), Harmonic Index Ratio (HIR) and Sum of Maximum Power (SMP) features

```
    if strcmp(dataset{i, 2}(5), 'A')
        RE_A = trapz(f, aPSD);
        MP_A = max(aPSD);
    else
        RE_B = trapz(f, aPSD);
        MP_B = max(aPSD);
        eKF(i-1).RE = RE_A ./ RE_B;
        eKF(i-1).HIR = eKF(i-1).HI_A ./ eKF(i-1).HI_B;
        eKF(i-1).SMP = MP_A + MP_B;
    end
```

Assign patient identifier (**ID**) and label (**PT**) as categorical variables

```
    eKF(i).SUBJ = categorical(strrep(cellstr(dataset{i, 2}(1:4)), 'SA', 'HS'));
    PT(i, 1) = categorical(strrep(cellstr(dataset{i, 2}(1:2)), 'SA', 'HS'));
    clear data btwfData pxx pyy pzz dM aPSD sPSD nPD RE_B MP_B sumPD posMedian ...
        lowPos highPos posPPF rArea
end
```

Convert **eKF** to table and add two new columns, **PDET** and **TPHS**. Each column will contain the labels needed to train and test the classification models.

```
eKF = struct2table(eKF);
eKF.PDET = PT;
eKF.TPHS = PT;
eKF.TPHS('PD' == eKF.TPHS(:) | 'ET' == eKF.TPHS(:)) = 'TP';
```

Create a new column named **Id** in which a number is assigned according to the label of the subject in the **PDET** column.

```
eKF.Id(1:length(eKF.SUBJ), 1) = 0;
eKF.Id(strcmp('PD', string(eKF.PDET(:)))) = 1;
eKF.Id(strcmp('ET', string(eKF.PDET(:)))) = 2;
eKF.Id(strcmp('HS', string(eKF.PDET(:)))) = 3;
eKF(cellfun('isempty', eKF{:, 'RE'}), :) = [];
eKF = struct2table(table2struct(eKF));
```

## Feature evaluation and selection

Data standardization, this process contributes to the performance of some classification methods.

```
eKF(:,1:13) = array2table(normalize(eKF{:,1:13}, 'zscore'));

for j = 1:2
```

Assign the labels to use as Successful Class (**sC**) and Failure Class (**fC**).

```
    if j == 1
        fC = 'HS';
        sC = {'PD', 'ET'};
        cC = 16;
    else
        fC = 'ET';
        sC = 'PD';
        cC = 15;
    end
    subj = cellstr(string([eKF.SUBJ]));
```

Find the indexes where the first two **SUBJ** chars are equal to TP and HS or PD and ET, respectively.

```
    idFC = find(string(cellfun(@(x)x(1:2), subj, 'UniformOutput', false)) == fC);
    idSC = find(logical(sum(string(cellfun(@(x)x(1:2), subj, 'UniformOutput', false)) ...
        == sC, 2)));
```

Setting up **predictors** and **response**

```
    predictors = eKF{[idSC; idFC],1:13};
    response = eKF{[idSC; idFC],cC};
```

Rank the predictors using chi-square tests and Umbiased trees.

```
    [idx,scores] = fscchi2(predictors,response);
    rankChi2 = false(1,length(idx));
    rankChi2(idx(1:5)) = true;

    [~,idx] = sortrows(predictorImportance(fitctree(predictors,response,...
        'PredictorSelection','curvature','Surrogate','on'))', 'descend');
    rankUmbiased = false(1, length(idx));
    rankUmbiased(idx(1:5)) = true;
```

Select the most important predictors.

```
    sKF = eKF.Properties.VariableNames;
    sKF = sKF(rankChi2  & rankUmbiased);
```

Assign the selected features to the respective variable (**sKF_TPHS** and **sKF_PDET**) according to the evaluated Case.

```
    if j == 1
        sKF_TPHS = sKF;

    else
        sKF_PDET = sKF;
    end
end
```

Save **eKF** as .mat file

```
save('kinematicFeatures.mat', 'eKF', 'sKF_TPHS', 'sKF_PDET');
```

# S2 Appendix. Matlab code for Model Training and Testing

```
clc;
clearvars;
close all;
warning('off', 'all');
```

Set the folder (**newFolder**) where the models will be stored.

```
rootFolder = 'D:\';
newFolder = [rootFolder, 'TrainedModels\'];
oldFolder = cd(newFolder);
```

Add **oldFolder** to the path to use all created functions

```
addpath(oldFolder);
```

Define the training/testing divisions

```
thresholdTrainTest = [0.3, 0.5, 0.7];
```

Create the variables **TM_TPHS** and **TM_PDET** that store the performance metrics of the models in each iteration.

```
for tt = 1:3
    TM_TPHS = [];
    TM_PDET = [];
```

Create the folder (**tM_Folder**) where the models will be saved. The name of the folder will be composed as follows: date (ddmmyyyyy) + TrainedModels + training/testing division (Example: 01062019TrainedModels7030)

```
    tM_Folder = [replace(datestr(now, 'dd/mm/yyyy'), '/', ''), 'TrainedModels',...
        num2str(thresholdTrainTest(tt)*100), num2str(100-thresholdTrainTest(tt)*100)];
    mkdir(tM_Folder);
    cd(tM_Folder)
```

Check if the **iP.mat** file exists, which contains the last iteration performed. If it does not exist, **n** is initialized in 1. This ensures that the process won't restart if for some reason it was interrupted.

```
    if exist([newFolder, tM_Folder, '\iP.mat'], 'file') == 2
        load('iP.mat');
        nf = 100;
        n = i + 1;
    else
        n = 1;
        nf = 100;
    end
```

Start iterations up to **nf** times

```
    for i = n:nf
```

Use the **RandomDivision** function to perform the random division. Training and testing sets are obtained from the TPHS and PDET groups.

```
        rng shuffle
        fC = 'HS'; sC = {'PD', 'ET'};
        [TPHS_train, TPHS_test, TPHS_sF] = RandomDivision(thresholdTrainTest, fC, sC);
        fC = 'ET'; sC = 'PD';
        [PDET_train, PDET_test, PDET_sF] = RandomDivision(thresholdTrainTest, fC, sC);
```

Create the folders where the models of each group are stored.

```
        mkdir('TPHS');
        fTPHS = [newFolder, tM_Folder, '\TPHS\'];
        mkdir('PDET');
        fPDET = [newFolder, tM_Folder, '\PDET\'];
```

Run the **MassiveTrain** function to train and test the classification models. The variables **TPHS_pm** and **PDET_pm** contain the performance metrics of the models. The variable **KFolds** is the number of folds used for cross-validation.

```
        KFolds = 5;
        TPHS_pm = MassiveTrain(TPHS_train, fTPHS, KFolds, TPHS_test, TPHS_sF, 'HS', 'TP');
        PDET_pm = MassiveTrain(PDET_train, fPDET, KFolds, PDET_test, PDET_sF, 'ET', 'PD');
```

Store the performance metrics

```
        TM_TPHS = cat(3, TM_TPHS, TPHS_pm);
        TM_PDET = cat(3, TM_PDET, PDET_pm);
```

Run the **ModelSelection** function to find the top 5 classification models (**Top5_TPHS** and **Top5_PDET**) for both groups based on their performance metrics.

```
        Top5_TPHS = ModelSelection(TM_TPHS);
        Top5_PDET = ModelSelection(TM_PDET);
        cd([newFolder, tM_Folder]);
```

Save the computed variables as .mat file into **tM_Folder** and return to **newFolder.**

```
        save('CMTPHS.mat', 'TM_TPHS', 'Top5_TPHS');
        save('CMPDET.mat', 'TM_PDET', 'Top5_PDET');
        save('iP.mat', 'i', 'tM_Folder');
        cd(newFolder);
    end
```

```
end
```

## Model Training and Testing functions

### RandomDivision function

```
function [trainingData, testingData, sF] = RandomDivision(thresholdTrainTest, fC, sC)
% Case 1: TP vs. HS. -> successfulClass (sC): Tremor Patients - TP.
%                       failureClass (fC): Healthy Subjects  - HS.
% Case 2: PD vs. ET. -> successfulClass (sC): Parkinson's Disease - PD.
%                       failureClass (sC): Essential Tremor - ET.

% Load the eKF table with the kinematic features and transform it in a struct.
load('kinematicFeatures.mat', 'eKF', 'sKF_TPHS', 'sKF_PDET');

% Assign to the variable sF the selected kinematic features according to the Case.
if strcmp(fC, 'HS') == 1
    sF = sKF_TPHS;
else
    sF = sKF_PDET;
end

% Extract the SUBJ column and convert it to cell.
subj = cellstr(string([eKF.SUBJ]));

% Find the indexes where the first two id chars are equal to TP and HS or PD and ET
% respectively.
idFC = find(string(cellfun(@(x)x(1:2), subj, 'UniformOutput', false)) == fC);
idSC = find(logical(sum(string(cellfun(@(x)x(1:2), subj, 'UniformOutput', false))...
    == sC, 2)));

% Find the number of classes.  The following FOR loop allows the assignment of the
% training and testing subjects at the same established ratio.
max_Int = max(eKF{[idSC; idFC], {'Id'}});
trainingData = [];
testingData = [];
eKF = table2struct(eKF);

for i = 1:max_Int
    idG = find([eKF.Id]' == i);

    % Use the randperm command to randomly distribute the indexes for each label.
    id_rand = idG(randperm(length(idG), (length(idG))));

    % Extract the proportion of data that will be used for training.
    sz_train = round(length(idG)*thresholdTrainTest);

    % Extract the training data.
    rand_train = eKF(contains(string(subj), string(subj(id_rand(1:sz_train), :))));

    % Extract the testing data.
    rand_test = eKF(contains(string(subj), string(subj(id_rand(sz_train+1:end), :))));

    % Create the training data table.
```

```
    trainingData = [trainingData; struct2table(rand_train)];%#ok<AGROW>

    % Create the testing data table.
    testingData = [testingData; struct2table(rand_test)];%#ok<AGROW>
    clear idG id_rand sz_train rand_train rand_test
end

% Create the training data table
trainingData.SUBJ = [];
if strcmp(fC, 'HS') == 1
    trainingData.PDET = [];
    trainingData.TPHS = removecats([trainingData.TPHS]);
else
    trainingData.TPHS = [];
    trainingData.PDET = removecats([trainingData.PDET]);
end

% Create the testing data table
testingData.SUBJ = [];
if strcmp(fC, 'HS') == 1
    testingData.PDET = [];
    testingData.TPHS = removecats([testingData.TPHS]);
else
    testingData.TPHS = [];
    testingData.PDET = removecats([testingData.PDET]);
end
end
```

## MassiveTrain function

```
function models = MassiveTrain(trainingData, dataFolder, KFolds, testingSet, sF, fC, sC)
% Initial setup.
sFLength = length(sF);
aTM = cell(25, 11);
models = [];

for i = 1:sFLength
    % Function C = nchoosek(v,i) returns an array which contains all the possible
    % combinations of the elements of the vector v taken i at once, that is, it takes only
    % once a combination of features.
    fComb = nchoosek(sF, i);
    pMetrics = [];

    for j = 1:length(fComb(:, 1))
        % Assign to variable FComb the combination of features that are in position j of
        % variables fComb.
        FComb = fComb(j, :);

        % Start parallel pool. The number of workers will depend on the number of cores in
        % the work station.
        p = gcp();

        % Run the trainClassifier function within the parfeval function to train and test
        % the models in parallel mode.
```

```
        for k = 1:25
            job(k) = parfeval(p, @trainClassifier, 1, trainingData, FComb, k, KFolds,...
                testingSet, fC, sC); %#ok<AGROW>
        end

        for k = 1:25
            % The fetchNext function returns the struct elements tM* which contain the
            % models and their performance metrics.
            tic
            [Id, tM] = fetchNext(job);
            trainTime = toc;
            tM.trainTime = trainTime;

            % Define the model name.
            featureN = char(replace(join(string(FComb)), {' '}, {'-'}));

            % Store the performance metrics in the aTM* cell elements.
            pM = struct2cell(rmfield(tM', {'trainedModel'}))';
            aTM(Id, :) = {[dataFolder, num2str(i), 'KF.mat'], pM{1}, pM{8}, featureN,...
                pM{7}, 'NO', pM{2}, pM{3}, pM{4}, pM{5}, pM{6}};
        end
        % Store the atM* cell elements in pMetrics
        pMetrics = [pMetrics; aTM]; %#ok<AGROW>
        save([dataFolder, num2str(i), 'KFpM.mat'], 'pMetrics');
        clear k FC atM  trainTime pM;
    end
    % Store pMetrics in models
    models = [models; pMetrics]; %#ok<AGROW>
    clear j;
end
end
```

## trainClassifier function

```
function [tM] = trainClassifier(trainingData, predictorNames, method, KFolds,...
    testingData, fC, sC)
% This code processes the data into the right shape for training the model.
% Extract response
inputTable = trainingData;
idResponse = find(strcmp(inputTable.Properties.VariableNames, strcat(sC,fC)));
response = inputTable.(idResponse);

% Data transformation: Select subset of the features and extract predictors.
predictors = inputTable(:, predictorNames);
isCategoricalPredictor = false(1,length(predictorNames));

% Train a classifier. This code specifies all the classifier options and trains the
% classifier.
if method == 6
    % For logistic regression, the response values must be converted to zeros and ones
    % because the responses are assumed to follow a binomial distribution.
    % 1 or true = 'successful' class & 0 or false = 'failure' class
    % NaN - missing response.
    successClass = sC;
```

```
    failureClass = fC;

    % Compute the majority response class. If there is a NaN-prediction from fitglm,
    % convert NaN to this majority class label.
    numSuccess = sum(response == successClass);
    numFailure = sum(response == failureClass);
    if numSuccess > numFailure
        missingClass = successClass;
    else
        missingClass = failureClass;
    end
    responseCategories = {successClass, failureClass};
    successFailureAndMissingClasses = categorical({successClass; failureClass;...
        missingClass}, responseCategories);
    isMissing = isundefined(response);
    zeroOneResponse = double(ismember(response, successClass));
    zeroOneResponse(isMissing) = NaN;
    clear numSuccess numFailure responseCategories isMissing;

    % Prepare input arguments to fitglm.
    concatenatedPredictorsAndResponse = [predictors, table(zeroOneResponse)];

    % Train using fitglm.
    Method = 'LogisticRegression';
    Model = fitglm(concatenatedPredictorsAndResponse, 'Distribution', 'binomial',...
        'link', 'logit');
    clear concatenatedPredictorsAndResponse zeroOneResponse;

    % Convert predicted probabilities to predicted class labels and scores.
    convertSuccessProbsToPredictions = @(p) successFailureAndMissingClasses(~isnan(p)...
        .*((p<0.5) + 1 ) + isnan(p)*3);
    returnMultipleValuesFcn = @(varargin) varargin{1:max(1,nargout)};
    scoresFcn = @(p) [p, 1-p];
    predictionsAndScoresFcn = @(p) returnMultipleValuesFcn...
        (convertSuccessProbsToPredictions(p), scoresFcn(p) );

    % Add additional fields to the result struct.
    trainedModel.SuccessClass = successClass;
    trainedModel.FailureClass = failureClass;
    trainedModel.MissingClass = missingClass;
    trainedModel.ClassNames = {successClass; failureClass};

    % Create the predict function.
    PredictFcn = @(x) predictionsAndScoresFcn(predict(Model, x));
    clear predictionsAndScoresFcn successFailureAndMissingClasses scoresFcn...
        successClass convertSuccessProbsToPredictions returnMultipleValuesFcn...
        failureClass missingClass;
elseif method == 7 || method == 8
    % Expand the Distribution Names per predictor. Numerical predictors are assigned
    % either Gaussian or Kernel distribution and categorical predictors are assigned mvmn
    % distribution Gaussian is replaced with Normal when passing to the fitcnb function.
    if method == 7
        Method = 'GaussianNaiveBayes';
        distribution = 'Kernel';
```

```
        Kernel = 'Normal';
        Support = 'Unbounded';
    else
        Method = 'KernelNaiveBayes';
        distribution = 'Normal';
        Kernel = [];
        Support = [];
    end
    distributionNames =  repmat({distribution}, 1, length(isCategoricalPredictor));
    distributionNames(isCategoricalPredictor) = {'mvmn'};

     % Train using fitcnb.
    Model = fitcnb(predictors, response, 'Kernel', Kernel, 'Support', Support,...
        'DistributionNames', distributionNames, 'ClassNames', categorical({sC; fC}));

    % Create the predict function.
    PredictFcn = @(x) predict(Model, x);
else
    % Train using the selected method.
    [Model, Method] = classMode(method, predictors, response, sC, fC);

    % Create the predict functions.
    PredictFcn = @(x) predict(Model, x);
end
% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
featureSelectionFcn = @(x) x(:,predictorNames);
trainedModel.predictFcn = @(x) PredictFcn(featureSelectionFcn(predictorExtractionFcn(x)));
clear PredictFcn

% Add additional fields to the result struct
trainedModel.RequiredVariables = predictorNames;
trainedModel.ClassificationModel = Model;
trainedModel.About = ['This struct is a trained model exported from Classification Lea'...
    'rner R2019b.'];
trainedModel.HowToPredict = sprintf(['To make predictions on a new table, T, use: \n  '...
    'yfit = c.predictFcn(T) \nreplacing ''c'' with the name of the variable that is th'...
    'is struct, e.g. ''trainedModel''. \n \nThe table, T, must contain the variables r'...
    'eturned by: \n  c.RequiredVariables \nVariable formats (e.g. matrix/vector, datat'...
    'ype) must match the original training data. \nAdditional variables are ignored. \'...
    'n \nFor more information, see <a href="matlab:helpview(fullfile(docroot, ''stats'...
    ''', ''stats.map''), ''appclassification_exportmodeltoworkspace'')">How to predict'...
    ' using an exported model</a>.']);

% Perform cross-validation
if method ~= 6
    partitionedModel = crossval(trainedModel.ClassificationModel, 'KFold', KFolds);

    % Compute validation predictions
    [validationPredictions, validationScores] = kfoldPredict(partitionedModel);

    % Compute validation accuracy
    cvAcc = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');
```

```
else
    cvp = cvpartition(response, 'KFold', KFolds);

    % Initialize the predictions to the proper sizes
    validationPredictions = response;
    numObservations = size(predictors, 1);
    numClasses = 2;
    validationScores = NaN(numObservations, numClasses);

    for fold = 1:KFolds
        trainingPredictors = predictors(cvp.training(fold), :);
        trainingResponse = response(cvp.training(fold), :);

        % Data transformation: Select subset of the features
        trainingPredictors = trainingPredictors(:,predictorNames);
        foldIsCategoricalPredictor = true(1,length(predictorNames));%#ok<NASGU>

        % Train a classifier. For logistic regression, the response values must be
        % converted to zeros and ones because the responses are assumed to follow a
        % binomial distribution.
        % 1 or true = 'successful' class & 0 or false = 'failure' class
        % NaN - missing response.
        successClass = sC;
        failureClass = fC;

        % Compute the majority response class. If there is a NaN-prediction from fitglm,
        % convert NaN to this majority class label.
        numSuccess = sum(trainingResponse == successClass);
        numFailure = sum(trainingResponse == failureClass);
        if numSuccess > numFailure
            missingClass = successClass;
        else
            missingClass = failureClass;
        end
        responseCategories = {successClass, failureClass};
        successFailureAndMissingClasses = categorical({successClass; failureClass;...
            missingClass}, responseCategories);
        isMissing = isundefined(trainingResponse);
        zeroOneResponse = double(ismember(trainingResponse, successClass));
        zeroOneResponse(isMissing) = NaN;

        % Prepare input arguments to fitglm.
        concatenatedPredictorsAndResponse = [trainingPredictors, table(zeroOneResponse)];

        % Train using fitglm.
        Model = fitglm(concatenatedPredictorsAndResponse, 'Distribution', 'binomial',...
            'link', 'logit');

        % Convert predicted probabilities to predicted class labels and scores.
        convertSuccessProbsToPredictions = @(p) successFailureAndMissingClasses...
            (~isnan(p).*((p<0.5) + 1 ) + isnan(p)*3);
        returnMultipleValuesFcn = @(varargin) varargin{1:max(1,nargout)};
        scoresFcn = @(p) [p, 1-p];
        predictionsAndScoresFcn = @(p) returnMultipleValuesFcn...
```

```
            (convertSuccessProbsToPredictions(p), scoresFcn(p));

        % Create the result struct with predict function
        featureSelectionFcn = @(x) x(:,predictorNames);
        PredictFcn = @(x) predictionsAndScoresFcn(predict(Model, x));
        validationPredictFcn = @(x) PredictFcn(featureSelectionFcn(x));

        % Compute validation predictions
        validationPredictors = predictors(cvp.test(fold), :);
        [foldPredictions, foldScores] = validationPredictFcn(validationPredictors);

        % Store predictions in the original order
        validationPredictions(cvp.test(fold), :) = foldPredictions;
        validationScores(cvp.test(fold), :) = foldScores;
        clear successFailureAndMissingClasses returnMultipleValuesFcn foldPredictions...
            convertSuccessProbsToPredictions scoresFcn PredictFcn validationPredictors...
            foldScores validationPredictFcn predictionsAndScoresFcn;
    end
    clear trainingResponse trainingPredictors;

    % Compute validation accuracy
    correctPredictions = (validationPredictions == response);
    isMissing = ismissing(response);
    correctPredictions = correctPredictions(~isMissing);
    cvAcc = sum(correctPredictions)/length(correctPredictions);
end

% Compute the performance metrics.
[tAcc, tSen, tSpe, TP, TN] = PerformanceCalculation(testingData, trainedModel);

% Create the result struct tM.
tM = struct('trainedModel', trainedModel, 'Acc', cvAcc, 'Acc_Test', tAcc, 'Sen_Test',...
    tSen, 'Spe_Test', tSpe, 'TP', TP, 'TN', TN, 'Method', Method);
end
```

## classMode function

```
function [trainedModel, Method] = classMode(method, predictors, response, sC, fC)
% Train using the selected method.
switch method
    case 1
        Method = 'ComplexTree';
        trainedModel = fitctree(predictors, response, 'SplitCriterion', 'gdi',...
            'MaxNumSplits', 100, 'Surrogate', 'off', 'ClassNames', categorical({sC; fC}));
    case 2
        Method = 'MediumTree';
        trainedModel = fitctree(predictors, response, 'SplitCriterion', 'gdi',...
            'MaxNumSplits', 20, 'Surrogate', 'off', 'ClassNames', categorical({sC; fC}));
    case 3
        Method = 'SimpleTree';
        trainedModel = fitctree(predictors, response, 'SplitCriterion', 'gdi',...
            'MaxNumSplits', 4, 'Surrogate', 'off', 'ClassNames', categorical({sC; fC}));
    case 4
        Method = 'LinearDiscriminant';
```

```
        trainedModel = fitcdiscr(predictors, response, 'DiscrimType', 'linear',...
            'Gamma', 0, 'FillCoeffs', 'off', 'ClassNames', categorical({sC; fC}));
    case 5
        Method = 'QuadraticDiscriminant';
        trainedModel = fitcdiscr(predictors, response, 'DiscrimType', 'diagQuadratic',...
            'FillCoeffs', 'off', 'ClassNames', categorical({sC; fC}));
    case 9
        Method = 'LinearSVM';
        trainedModel = fitcsvm(predictors, response, 'KernelFunction', 'linear',...
            'PolynomialOrder', [], 'KernelScale', 'auto', 'BoxConstraint', 1,...
            'Standardize', true, 'ClassNames', categorical({sC; fC}));
    case 10
        Method = 'QuadraticSVM';
        trainedModel = fitcsvm(predictors, response, 'KernelFunction', 'polynomial',...
            'PolynomialOrder', 2, 'KernelScale', 'auto', 'BoxConstraint', 1,...
            'Standardize', true, 'ClassNames', categorical({sC; fC}));
    case 11
        Method = 'CubicSVM';
        trainedModel = fitcsvm(predictors, response, 'KernelFunction', 'polynomial',...
            'PolynomialOrder', 3, 'KernelScale', 'auto', 'BoxConstraint', 1,...
            'Standardize', true, 'ClassNames', categorical({sC; fC}));
    case 12
        Method = 'FineGaussianSVM';
        trainedModel = fitcsvm(predictors, response, 'KernelFunction', 'gaussian',...
            'PolynomialOrder', [], 'KernelScale', 0.25, 'BoxConstraint', 1,...
            'Standardize', true, 'ClassNames', categorical({sC; fC}));
    case 13
        Method = 'MediumGaussianSVM';
        trainedModel = fitcsvm(predictors, response, 'KernelFunction', 'gaussian', ...
            'PolynomialOrder', [], 'KernelScale', 1, 'BoxConstraint', 1, 'Standardize',...
            true, 'ClassNames', categorical({sC; fC}));
    case 14
        Method = 'CoarseGaussianSVM';
        trainedModel = fitcsvm(predictors, response, 'KernelFunction', 'gaussian', ...
            'PolynomialOrder', [], 'KernelScale', 4, 'BoxConstraint', 1, 'Standardize',...
            true, 'ClassNames', categorical({sC; fC}));
    case 15
        Method = 'FineKNN';
        trainedModel = fitcknn(predictors, response, 'Distance', 'Euclidean',...
            'Exponent', [], 'NumNeighbors', 1, 'DistanceWeight', 'Equal',...
            'Standardize', true, 'ClassNames', categorical({sC; fC}));
    case 16
        Method = 'MediumKNN';
        trainedModel = fitcknn(predictors, response, 'Distance', 'Euclidean',...
            'Exponent', [], 'NumNeighbors', 10, 'DistanceWeight', 'Equal',...
            'Standardize', true, 'ClassNames', categorical({sC; fC}));
    case 17
        Method = 'CoarseKNN';
        trainedModel = fitcknn(predictors, response, 'Distance', 'Euclidean',...
            'Exponent', [], 'NumNeighbors', 100, 'DistanceWeight', 'Equal',...
            'Standardize', true, 'ClassNames', categorical({sC; fC}));
    case 18
        Method = 'CosineKNN';
        trainedModel = fitcknn(predictors, response, 'Distance', 'Cosine',...
```

```
            'Exponent', [], 'NumNeighbors', 10, 'DistanceWeight', 'Equal',...
            'Standardize', true, 'ClassNames', categorical({sC; fC}));
    case 19
        Method = 'CubicKNN';
        trainedModel = fitcknn(predictors, response, 'Distance', 'Minkowski',...
            'Exponent', 3, 'NumNeighbors', 10, 'DistanceWeight', 'Equal',...
            'Standardize', true, 'ClassNames', categorical({sC; fC}));
    case 20
        Method = 'WeightedKNN';
        trainedModel = fitcknn(predictors, response, 'Distance', 'Euclidean',...
            'Exponent', [], 'NumNeighbors', 10, 'DistanceWeight', 'SquaredInverse',...
            'Standardize', true, 'ClassNames', categorical({sC; fC}));
    case 21
        Method = 'EnsembleBoostedTrees';
        template = templateTree('MaxNumSplits', 20);
        trainedModel = fitcensemble(predictors, response, 'Method', 'AdaBoostM1',...
            'NumLearningCycles', 30, 'Learners', template, 'LearnRate', 0.1,...
            'ClassNames', categorical({sC; fC}));
    case 22
        Method = 'EnsembleBaggedTrees';
        template = templateTree('MaxNumSplits', 34);
        trainedModel = fitcensemble(predictors, response, 'Method', 'Bag',...
            'NumLearningCycles', 30, 'Learners', template, 'ClassNames',...
            categorical({sC; fC}));
    case 23
        Method = 'EnsembleSubspaceDiscriminant';
        subspaceDimension = max(1, min(1, width(predictors)-1));
        trainedModel = fitcensemble(predictors, response, 'Method', 'Subspace',...
            'NumLearningCycles', 30, 'Learners', 'discriminant', 'NPredToSample',...
            subspaceDimension, 'ClassNames', categorical({sC; fC}));
    case 24
        Method = 'EnsembleSubspaceKNN';
        subspaceDimension = max(1, min(1, width(predictors)-1));
        trainedModel = fitcensemble(predictors, response, 'Method', 'Subspace',...
            'NumLearningCycles', 30, 'Learners', 'knn', 'NPredToSample',...
            subspaceDimension, 'ClassNames', categorical({sC; fC}));
    case 25
        Method = 'EnsembleRUSBoostedTrees';
        template = templateTree('MaxNumSplits', 20);
        trainedModel = fitcensemble(predictors, response, 'Method', 'RUSBoost',...
            'NumLearningCycles', 30, 'Learners', template, 'LearnRate', 0.1,...
            'ClassNames', categorical({sC; fC}));
end
clear method predictors response sC fC;
end
```

## PerformanceCalculation function

```
function [Acc, Sen, Spe, TP, TN] = PerformanceCalculation(testingnData, trainedModel)
% Case 1: TP vs. HS. -> successfulClass (sC): Tremor Patients - TP.
%                       failureClass (fC): Healthy Subjects  - HS.
% Case 2: PD vs. ET. -> successfulClass (sC): Parkinson's Disease - PD.
%                       failureClass (sC): Essential Tremor - ET.
```

```
% Assign testingnData to dTest and use it to compute the trainedModel predictions.
dTest = testingnData;
yFit = trainedModel.predictFcn(dTest);

idResponse = find(strcmp(dTest.Properties.VariableNames, 'PDET')...
    | strcmp(dTest.Properties.VariableNames, 'TPHS'));

% Identify the successful and failure class based on the case.
if strcmp(string(dTest.Properties.VariableNames(idResponse)), 'PDET') == 1
    yFitSC = yFit([dTest.Id] == 1, 1);
    dTestSC = dTest([dTest.Id] == 1, idResponse).PDET;
    yFitFC = yFit([dTest.Id] == 2, 1);
    dTestFC = dTest([dTest.Id] == 2, idResponse).PDET;
    tP = sum([dTest.Id] == 1);
    tN = sum([dTest.Id] == 2);
else
    yFitSC = yFit([dTest.Id] == 1 | [dTest.Id] == 2, 1);
    dTestSC = dTest([dTest.Id] == 1 | [dTest.Id] == 2, idResponse).TPHS;
    yFitFC = yFit([dTest.Id] == 3, 1);
    dTestFC = dTest([dTest.Id] == 3, idResponse).TPHS;
    tP = sum([dTest.Id] == 1 | [dTest.Id] == 2);
    tN = sum([dTest.Id] == 3);
end

% Compute the successful and failure class correct predictions.
sC_Predictions = sum(yFitSC == dTestSC);
fC_Predictions = sum(yFitFC == dTestFC);

% Compute Sensitivity, Specificity, and Accuracy.
TP = sC_Predictions;
FN = abs(tP-TP);
Sen = (TP / (TP + FN));
TN = fC_Predictions;
FP = abs(tN-TN);
Spe = (TN / (TN + FP));
Acc = (TP + TN) / (TP + TN + FP + FN);
end
```

## ModelSelection function

```
function TF = ModelSelection(trainedModels)

% Compute the average values and their standard deviation from the metrics obtained by
% the models across the 100 iterations.
scoreTM(:, 1) = trainedModels(:, 1, 1);
scoreTM(:, 2) = num2cell(nanmean(cell2mat(trainedModels(:, 2, :)).*100,3));
scoreTM(:, 3) = num2cell(nanmean(cell2mat(trainedModels(:, 7, :)).*100,3));
scoreTM(:, 4) = num2cell(nanstd(cell2mat(trainedModels(:, 7, :)).*100,0,3));
scoreTM(:, 5) = num2cell(nanmean(cell2mat(trainedModels(:, 8, :)).*100,3));
scoreTM(:, 6) = num2cell(nanstd(cell2mat(trainedModels(:, 8, :)).*100,0,3));
scoreTM(:, 7) = num2cell(nanmean(cell2mat(trainedModels(:, 9, :)).*100,3));
scoreTM(:, 8) = num2cell(nanstd(cell2mat(trainedModels(:, 9, :)).*100,0,3));
scoreTM(:, 9) = num2cell(nanmean(cell2mat(trainedModels(:, 3, :)).*1000,3));
scoreTM(:, 10) = num2cell(nanstd(cell2mat(trainedModels(:, 3, :)).*100,0,3));
```

```
scoreTM(:, 11) = trainedModels(:, 4, 1);
scoreTM(:, 12) = trainedModels(:, 5, 1);

% Sort out the models from highest to lowest performance
TM_sort = sortrows(scoreTM, [3, 5, 7, 2, 9], {'descend', 'descend', 'descend',...
    'descend', 'ascend'});
Top3_TM = TM_sort(1:5, :);

% Find the 3 best models. All the models have a cell element in which the metrics
% obtained in the n iterations will be.
[r, ~] = size(Top3_TM);
TF = cell(3,14);

for i = 1:r
    comp_TM = trainedModels(strcmp(Top3_TM{i, 11}, trainedModels(:, 4, 1)) &...
        strcmp(Top3_TM{i, 12}, trainedModels(:, 5, 1)), :, :);

    Top3_TM{i, 13} = reshape(permute(comp_TM, [1, 3, 2]), [], size(comp_TM, 2), 1);
    TF(i,[1:3,5,6,8,9,11,12,14]) = Top3_TM(i,[11,12,3:10]);
    TF(i,[4,7,10,13]) = {char(177)};
end
end
```

# Appendix D

# Python Code: *Data Analysis*

# Python code for Data Analysis

September 19, 2023

Import necessary libraries and modules.

[ ]:
```
import os
from tkinter import filedialog
import tkinter as tk

import numpy as np
import pandas as pd
from scipy.integrate import simps
from scipy.signal import welch, butter, filtfilt
```

Function to calculate features from **Power Spectral Density (PSD)**

[ ]:
```
def calculate_features(psd, frequencies):
    """
    Calculates features from power spectral density (PSD).

    Args:
    psd (array-like): Power spectral density values.
    frequencies (array-like): Corresponding frequencies.

    Returns:
    tuple: A tuple containing the calculated features.
    """
    bp = simps(psd, frequencies)
    psd_normalized = psd / bp
    cumulative_sum = np.cumsum(psd_normalized)
    half_total_power = 0.5
    mpf = frequencies[np.where(cumulative_sum >= half_total_power)[0][0]]

    frequency_5 = frequencies[np.where(cumulative_sum >= 0.05)[0][0]]
    frequency_95 = frequencies[np.where(cumulative_sum >= 0.95)[0][0]]
    pbw = frequency_95 - frequency_5

    pp_pos = np.argmax(psd)
    ppf = frequencies[pp_pos]
    pp = psd[pp_pos]
```

```
    return bp, mpf, pbw, ppf, pp
```

Function to calculate the **Harmonic Index (HI)**

[ ]:
```
def calculate_hi(psd, frequencies, ppf):
    """
    Calculates the Harmonic Index (HI) from power spectral density (PSD).

    Args:
    psd (array-like): Power spectral density values.
    frequencies (array-like): Corresponding frequencies.
    ppf (float): Peak frequency.
    low_idx (int): Index of the lower frequency bound.
    high_idx (int): Index of the higher frequency bound.

    Returns:
    float: The calculated Harmonic Index (HI).
    """
    area_psd = simps(psd, frequencies)
    rectangle_area = simps(np.full(len(psd), ppf, dtype=float), frequencies)
    hi = area_psd / rectangle_area
    return hi
```

Function to calculate the **Relative Power Contribution (RPC)**

[ ]:
```
def calculate_rpc(psd, frequencies):
    """
    Calculates the Relative Power Contribution (RPC) from power spectral␣
 ↪density (PSD).

    Args:
    psd (array-like): Power spectral density values.
    frequencies (array-like): Corresponding frequencies.

    Returns:
    float: The calculated Relative Power Contribution (RPC).
    """
    pb = simps(psd, frequencies)
    psd_normalized = psd / pb
    cumuluspsd = np.cumsum(psd_normalized)
    f1_idx = np.where(cumuluspsd >= 0.95)[0][0]
    psd_harmonics = simps(psd_normalized[f1_idx:], frequencies[f1_idx:])
    psd_total = simps(psd_normalized, frequencies)
    rpc = psd_harmonics / psd_total
    return rpc
```

Function to calculate **RE, HIR and SMP features**

[ ]:
```
def calculate(axis, features_df, sensor):
    features_df[f'{sensor}_{axis}re'] = features_df[f'{sensor}_{axis}bp_R'] /␣
↪features_df[f'{sensor}_{axis}bp_P']
    features_df[f'{sensor}_{axis}hir'] = features_df[f'{sensor}_{axis}hi_R'] /␣
↪features_df[f'{sensor}_{axis}hi_P']
    features_df[f'{sensor}_{axis}smp'] = features_df[f'{sensor}_{axis}pp_R'] /␣
↪features_df[f'{sensor}_{axis}pp_P']
    return features_df
```

Function to process sensor data and calculate features

[ ]:
```
def process_sensor_data(data, fs, sensor_name, subject, label, position):
    """
    Processes sensor data and calculates features.

    Args:
    data (pandas DataFrame): Sensor data.
    fs (float): Sampling frequency.
    window_size (int): Window size in seconds.
    sensor_name (str): Name of the sensor.

    Returns:
    dict: A dictionary containing the calculated features.
    """
    data -= data.mean()
    segment_duration = int(20 * fs)
    # Calculate the start and end index of the 20-second range centered in the␣
↪middle
    start_index = int(len(data) // 2 - segment_duration // 2)
    end_index = start_index + segment_duration

    # Extract the 20-second range of data
    ranged_data = data.iloc[start_index:end_index]

    low_cut, high_cut = 3.0, 12.0
    nyquist = 0.5 * fs
    low = low_cut / nyquist
    high = high_cut / nyquist
    b, a = butter(5, [low, high], btype='band')

    filtered_data = filtfilt(b, a, ranged_data, axis=0)

    # Sliding window method
    window_duration = 20  # segundos
    overlap = 0.00  # 50% de solapamiento

    # Calcular el tamaño de la ventana y el desplazamiento
```

```
    window_size = int(window_duration * fs)  # sample_rate es la frecuencia de␣
↪muestreo de los datos
    step_size = int(window_size * (1 - overlap))

    # Aplicar el método de aumento de datos
    segments = []
    start = 0
    while start + window_size <= len(filtered_data):
        segment = filtered_data[start:start + window_size]
        segments.append(segment)
        start += step_size

    feature_names = ['bp', 'mpf', 'pbw', 'ppf', 'pp', 'hi', 'rpc']
    features = {f'{sensor_name}_{axis}{feature}_{position}': [] for axis in␣
↪['', 'x_', 'y_', 'z_'] for feature in
                feature_names}

    for i, segment in enumerate(segments):
        frequencies, psd_segment = welch(np.transpose(segment), fs=fs,␣
↪window='hamming', nperseg=100, nfft=30000,
                                         scaling='density', detrend=False)
        psd_segment = np.transpose(psd_segment)

        dif_low = np.abs(frequencies - low_cut)
        dif_high = np.abs(frequencies - high_cut)
        low_idx = np.where(dif_low == dif_low.min())[0][0]
        high_idx = np.where(dif_high == dif_high.min())[0][0]

        for axis, psd_values in zip(['', 'x_', 'y_', 'z_'],
                                    [psd_segment.mean(axis=1)] + [psd_segment[:
↪, i] for i in range(3)]):
            bp, mpf, pbw, ppf, pp = calculate_features(psd_values[low_idx:
↪high_idx + 1],
                                                       frequencies[low_idx:
↪high_idx + 1])
            features[f'{sensor_name}_{axis}bp_{position}'].append(bp)
            features[f'{sensor_name}_{axis}mpf_{position}'].append(mpf)
            features[f'{sensor_name}_{axis}pbw_{position}'].append(pbw)
            features[f'{sensor_name}_{axis}ppf_{position}'].append(ppf)
            features[f'{sensor_name}_{axis}pp_{position}'].append(pp)
            features[f'{sensor_name}_{axis}hi_{position}'].append(
                calculate_hi(psd_values[low_idx:high_idx + 1],␣
↪frequencies[low_idx:high_idx + 1], ppf))
            features[f'{sensor_name}_{axis}rpc_{position}'].append(
                calculate_rpc(psd_values[low_idx:high_idx + 1],␣
↪frequencies[low_idx:high_idx + 1]))
```

```
        features_df = pd.DataFrame(features)
        features_df['subject'] = subject
        features_df['label'] = label
    features_df['segment'] = features_df['subject'] + '_s' + features_df.index.
↪astype(str)

    return features_df.reset_index(drop=True)
```

Function to process a directory and calculate features for all files

[ ]:
```
def process_directory(directory, device):
    """
    Processes a directory and calculates features for all files.

    Args:
    directory (str): Directory path.

    Returns:
    pandas DataFrame: The calculated features.
    """

    column_names = ['subject', 'label', 'position', 'segment']
    feature_names = ['bp', 'mpf', 'pbw', 'ppf', 'pp', 'hi', 'rpc']

    accelerometer_columns = [f'acc_{axis}{feature}_{position}' for position in␣
↪['R', 'P'] for axis in
                             ['', 'x_', 'y_', 'z_'] for feature in␣
↪feature_names]
    gyroscope_columns = [f'gyr_{axis}{feature}_{position}' for position in␣
↪['R', 'P'] for axis in ['', 'x_', 'y_', 'z_']
                         for feature in feature_names]

    column_names += accelerometer_columns + gyroscope_columns

    features_df = pd.DataFrame(columns=column_names)

    features_df = process_each_file(directory, device, features_df)

    return features_df
```

[ ]:
```
def select_directory():
    """
    Opens a file dialog to select a directory.

    Returns:
    str: The selected directory path.
```

```
    """
    root = tk.Tk()
    root.withdraw()
    directory = filedialog.askdirectory()
    root.destroy()
    return directory
```

[ ]:
```
def receive_directory_input():
    """
    Receive directory and device inputs from the user.

    Returns:
    list[Tuple[str, str]]: A list of tuples, each containing a directory path␣
 ↪and device type.
    """
    selected_directories = []

    while True:
        # Ask the user to select a directory
        directory = select_directory()
        if not directory:
            break  # If no directory is selected, break the loop

        device = get_device_info()
        selected_directories.append((directory, device))  # Add the directory␣
 ↪and device type to the list

    if not selected_directories:
        raise Exception("No directories selected.")

    return selected_directories
```

[ ]:
```
def process_all_directories(dir_device_pairs):
    """
    Process all directories and combine the results.

    Args:
    dir_device_pairs (list[Tuple[str, str]]): A list of tuples, each containing␣
 ↪a directory path and device type.

    Returns:
    pandas DataFrame: The combined features DataFrame from all directories.
    """
    features_dfs = []

    for directory, device in dir_device_pairs:
        features_df = process_directory(directory, device)
```

```
        features_dfs.append(features_df)

    # Combine the feature DataFrames from all directories
    combined_features_df = pd.concat(features_dfs, ignore_index=True)

    for axis in ['', 'x_', 'y_', 'z_']:
        combined_features_df = calculate(axis, combined_features_df, 'acc')
        combined_features_df = calculate(axis, combined_features_df, 'gyr')

    del combined_features_df['subject']
    del combined_features_df['position']
    del combined_features_df['segment']

    return combined_features_df
```

[ ]:
```
def get_device_info():
    """Ask for device input from user."""
    while True:
        device = input("Enter the device for the directory (1 for iPhone5s, 2␣
 ↪for Xsens DOT, q to Quit): ")
        if device.lower() == 'q':
            return None
        if device not in ['1', '2']:
            print('Wrong input. Please enter a correct input.')
            continue
        else:
            return device
```

[ ]:
```
def process_each_file(directory, device, features_df):
    """Process each file in the directory."""
    for file in os.listdir(directory):
        if file.endswith(".csv"):
            label = file[:2]
            subject = file[:4]
            position = 'R' if file[4] == 'A' else 'P'

            fs, cols, accelerometer_data, gyroscope_data =␣
 ↪get_device_specific_data(directory, file, device)

            accelerometer_subsegment_df =␣
 ↪process_sensor_data(accelerometer_data, fs, 'acc', subject,
                                                                label, position)
            gyroscope_subsegment_df = process_sensor_data(gyroscope_data, fs,␣
 ↪'gyr', subject, label,
                                                           position)

            joined_df = accelerometer_subsegment_df.join(
```

```
                gyroscope_subsegment_df.set_index(['segment', 'label',␣
↪'subject']),
                on=['segment', 'label', 'subject'])

            if position == 'R':
                joined_r = joined_df
            else:
                joined_p = joined_r.join(joined_df.set_index(['segment',␣
↪'label', 'subject']),
                                         on=['segment', 'label', 'subject'])
                features_df = pd.concat([features_df, joined_p],␣
↪ignore_index=True)

    return features_df
```

[ ]:
```
def get_device_specific_data(directory, file, device):
    """Get data specific to the device."""
    if device == '1':
        fs = 100.0
        cols = ['motionRotationRateX', 'motionRotationRateY',␣
↪'motionRotationRateZ',
                'motionUserAccelerationX', 'motionUserAccelerationY',␣
↪'motionUserAccelerationZ']
        data = pd.read_csv(os.path.join(directory, file), usecols=cols)

        accelerometer_data = data[['motionUserAccelerationX',␣
↪'motionUserAccelerationY', 'motionUserAccelerationZ']]
        gyroscope_data = data[['motionRotationRateX', 'motionRotationRateY',␣
↪'motionRotationRateZ']]

    elif device == '2':
        fs = 120.0
        cols = ['Gyr_X', 'Gyr_Y', 'Gyr_Z', 'Acc_X', 'Acc_Y', 'Acc_Z']
        data = pd.read_csv(os.path.join(directory, file), skiprows=7,␣
↪usecols=cols)

        accelerometer_data, gyroscope_data = transform_data(data)

    return fs, cols, accelerometer_data, gyroscope_data
```

[ ]:
```
def transform_data(data):
    """Transform the data based on device."""
    accelerometer_data = data[['Acc_X', 'Acc_Y', 'Acc_Z']]
    gyroscope_data = data[['Gyr_X', 'Gyr_Y', 'Gyr_Z']]

    accelerometer_data = accelerometer_data.copy()
```

```
    accelerometer_data['Acc_X'] = accelerometer_data['Acc_X'].multiply(0.
↪1019716).subtract(1)
    accelerometer_data['Acc_Y'] = accelerometer_data['Acc_Y'].multiply(0.
↪1019716).subtract(1)
    accelerometer_data['Acc_Z'] = accelerometer_data['Acc_Z'].multiply(0.
↪1019716).subtract(1)

    gyroscope_data = gyroscope_data.copy()
    gyroscope_data['Gyr_X'] = gyroscope_data['Gyr_X'].multiply(0.0174533)
    gyroscope_data['Gyr_Y'] = gyroscope_data['Gyr_Y'].multiply(0.0174533)
    gyroscope_data['Gyr_Z'] = gyroscope_data['Gyr_Z'].multiply(0.0174533)

    return accelerometer_data, gyroscope_data
```

[ ]:
```
def main():
    try:
        # Step 1 - Receive directory and device input from user
        selected_directories = receive_directory_input()

        # Step 2 - Process all directories and combine results
        combined_features_df = process_all_directories(selected_directories)

        # Step 3 - Save the combined features DataFrame to a .csv file
        output_filename = 'features.csv'
        combined_features_df.to_csv(output_filename, index=False)

        print(f"Features saved to {output_filename}")

    except Exception as e:
        print("An error occured: ", str(e))
```

Program execution starts here

[ ]:
```
if __name__ == '__main__':
    main()
```

# Appendix E

# Python Code: *Model Training & Testing*

# Python code for Model Training and Testing

September 19, 2023

Import necessary libraries and modules.

[ ]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from pycaret.classification import *
from copy import copy
```

Prepare parameters function.

[ ]:
```
def prepare_params():
    """Prepare PyCaret experiment parameters."""
    parameters = {
        "data": training_data,
        "target": 'label',
        "normalize": True,
        "feature_selection": True,
        "train_size": 0.8,
        "data_split_stratify": True,
        "fold_strategy": 'stratifiedkfold',
        "fold": 5,
        "n_jobs": -1
    }
    return parameters
```

Function for tuning and ensembling a model.

[ ]:
```
def tune_and_ensemble_model(model):
    """Tune, bag and boost the given model."""
    # Tuning
    tuned_model = tune_model(model, optimize=OPTIMIZE_METRIC,␣
 ↪choose_better=True,
                             return_train_score=True, n_iter=N_ITER)
    # Bagging
    bagged_model = ensemble_model(tuned_model, optimize=OPTIMIZE_METRIC,␣
 ↪choose_better=True,
                                  method='Bagging', return_train_score=True)
    # Boosting
```

```
    boosted_model = ensemble_model(tuned_model, optimize=OPTIMIZE_METRIC,␣
↪choose_better=True,
                                   method='Boosting', return_train_score=True)

    return tuned_model, bagged_model, boosted_model
```

Function to blend models.

[ ]:
```
def blend_models_sets(model_sets, method_index):
    """Blend models based on given method_index."""
    return blend_models([m_set[method_index] for m_set in model_sets],
                        optimize=OPTIMIZE_METRIC,
                        choose_better=True,
                        return_train_score=True)
```

Function to evaluate and finalize a model.

[ ]:
```
def evaluate_and_finalize_models(model, model_name):
    """Evaluate and finalize a model then make predictions."""
    print(f"\n{model_name} Evaluation:")
    evaluate_model(model)

    print(f"\n{model_name} Finalization and Predictions:")
    final_model = finalize_model(model)
    predictions = predict_model(final_model, data=unseen_data)

    return final_model, predictions
```

Load and filter dataset.

[ ]:
```
dataset = pd.read_csv('/content/drive/My Drive/features.csv')
dataset = dataset[dataset['label'] != 'HS'].reset_index(drop=True)
```

Separating features and target.

[ ]:
```
X = dataset.drop('label', axis=1)
y = dataset['label']
```

Split dataset into training and testing sets.

[ ]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.15,␣
↪stratify=y)
```

Create training_data and unseen_data dataframes.

[ ]:
```
training_data = X_train.copy()
training_data['label'] = y_train
unseen_data = X_test.copy()
unseen_data['label'] = y_test
```

Print shapes of Training and Unseen Data.

[ ]:
```
print(f'Shape of Training Data: {training_data.shape}')
print(f'Shape of Unseen Data : {unseen_data.shape}')
```

Set PyCaret experiment.

[ ]:
```
setup_params = prepare_params()
exp = setup(**setup_params)
```

Set optimization parameters.

[ ]:
```
OPTIMIZE_METRIC = 'F1'
N_ITER = 50
```

Select top 5 models, tune, ensemble then blend each selected model.

[ ]:
```
print("Selecting top 5 models based on optimization metric:")
top_models = compare_models(n_select=5, sort=OPTIMIZE_METRIC)

print("Tuning, ensembling, and blending each selected model:")
model_sets = [tune_and_ensemble_model(model) for model in top_models]
blends = [blend_models_sets(model_sets, i) for i in range(3)]
```

Identify the best model.

[ ]:
```
print("Identifying the best model based on the optimization metric:")
best_model = automl(optimize=OPTIMIZE_METRIC)
```

Calibrate the best model.

[ ]:
```
calibrated_best_model = calibrate_model(best_model)
```

Evaluate and finalize the best model and the calibrated best model.

[ ]:
```
print("Evaluating and finalizing the best model along with the calibrated␣
 ↪configuration:")
best_models_info = []
best_models_info.append(evaluate_and_finalize_models(best_model, "Best Model"))
best_models_info.append(evaluate_and_finalize_models(calibrated_best_model,␣
 ↪"Calibrated Best Model"))
```

# Appendix F

# Journal Paper: *TremorSoft: A decision support application for differential diagnosis between Parkinsons disease and Essential Tremor*

SoftwareX 22 (2023) 101393

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

# TremorSoft: An decision support application for differential diagnosis between Parkinson's disease and essential tremor

Julián D. Loaiza Duque [a], Antonio J. Sánchez Egea [a,*], Hernán A. González Rojas [a], Pedro Chaná-Cuevas [b], Joaquim J. Ferreira [c,d], João Costa [c]

[a] Department of Mechanical Engineering, Universitat Politècnica de Catalunya, Barcelona, Spain
[b] Centro de Trastornos del Movimiento (CETRAM), Facultad de Ciencias Médicas, Universidad de Santiago de Chile, Santiago de Chile, Chile
[c] Laboratory of Clinical Pharmacology and Therapeutics, Faculdade de Medicina, Universidade de Lisboa, Lisboa, Portugal
[d] CNS - Campus Neurológico Sénior, Torres Vedras, Portugal

ARTICLE INFO

*Article history:*
Received 5 July 2022
Received in revised form 29 December 2022
Accepted 20 April 2023

*Keywords:*
Mobile application
Machine learning
Inertial sensor
Tremor assessment

ABSTRACT

A cost-effective, non-invasive, and easy-to-use tool is presented that uses the 6-axis inertial sensor of the smartphone or a specific wearable sensor, boosted by machine learning, to support early differential diagnosis of Parkinson's disease and Essential Tremor. A dedicated web server helps extract the kinematic indexes from the recorded signals, implement the machine learning models and return the resulting classification to the App. Thus, clinicians can use this App as a support tool in the clinic, contributing to performing motor evaluations in the uncertain and undecided stages of the diseases and promoting appropriate, fast, and timely therapeutic responses.



## Code metadata

| | |
|---|---|
| Current code version | v1.0 |
| Permanent link to this code version | https://github.com/ElsevierSoftwareX/SOFTX-D-22-00185 |
| Legal Code License | Safe creative, ID: 2206021281741 |
| Code versioning system used | Git |
| Software code languages, tools, and services used | Kotlin, Android SDK, IntelliJ IDEA |
| Compilation requirements, operating environments & dependencies | IntelliJ IDEA or Android Studio, Android SDK, |
| If available link to developer documentation/manual | None |
| Support email for questions | julian.david.loaiza@upc.edu |

## Software metadata

| | |
|---|---|
| Current software version | v1.0 |
| Permanent link to executables of this version | |
| Legal Software License | Safe Creative, ID: 2206021281741 |
| Computing platform/Operating System | Android |
| Installation requirements & dependencies | Android 8.0 or higher |
| User manual | In the appendix |
| Support email for questions | julian.david.loaiza@upc.edu |

## 1. Motivation and significance

Essential Tremor (ET) is a neurological disorder characterized by the manifestation of involuntary and rhythmic tremors in different parts of the body and is considered to be the most common tremor disorder worldwide [1,2], primarily affecting older adults. In some cases, ET is confused with Parkinson's Disease (PD) [3], a progressive neurodegenerative disease that, among

* Corresponding author.
*E-mail addresses:* julian.loaiza.duque@upc.edu (Julián D. Loaiza Duque), antonio.egea@upc.edu (Antonio J. Sánchez Egea), hernan.gonzalez@upc.edu (Hernán A. González Rojas), pedro.chana@usach.cl (Pedro Chaná-Cuevas), joaquimjferreira@gmail.com (Joaquim J. Ferreira), jncosta@fm.ul.pt (João Costa).

https://doi.org/10.1016/j.softx.2023.101393

its clinical manifestations, and as in ET, affects movement. The difference between these conditions has both therapeutic and prognostic implications. Therefore, the early and correct differential diagnosis of PD and ET is crucial to ensure adequate and timely treatments to control the evolution of the disease and also, to prevent the patient from suffering adverse effects when receiving medication prescribed due to an erroneous diagnosis [4]. The confusion in diagnosis is caused mainly by the fact that, although these are two distinct diseases, there are similarities in physiological and psychological symptoms between PD and ET. This is why differentiation of the two nervous system disorders remains a challenging task even for movement disorder specialists [5]. Neuroimaging by single-photon emission computed tomography (SPECT) has been considered a potential diagnostic option to differentiate PD from ET [6]. However, it is a high-cost test that limits its use in developed countries and involves long waiting periods in countries where the test is available. In addition, the procedure is invasive and requires patient compatibility with the radiopharmaceutical tracer used for this test, which may limit its applicability [7]. Therefore, a medical evaluation by a neurologist remains the gold standard for diagnosing both diseases.

To date, there are no low-cost, easy-to-use confirmatory or diagnostic tests or tools that ensure accurate, safe and reproducible differentiation between the two diseases [8]. However, with the progressive development of smartphone technologies and functionalities in recent years, mobile applications have been created for the medical sector, some of them aimed at supporting the diagnosis, assessment, analysis and monitoring of some movement disorders [9,10]. However, the applications developed as a diagnostic support tool mainly focus on differentiating a pathological tremor from a physiological one, mainly for PD. The above has motivated researchers to be interested in developing new support systems for the differential diagnosis of different movement disorders, particularly differentiating PD from ET. Numerous methods using various approaches have been developed over the past two decades [11]. However, only a few systems have been implemented as ready-to-use applications until now. In one of our previous works, hand tremor recordings from subjects with a confirmed diagnosis of PD and ET were analyzed from a protocol described in [12] to analyze the recorded signals and extract biomarkers that could be used in the context of routine clinical care to support the differential diagnosis of these tremor disorders. The proposed protocol consists of an easy, quick test that requires no specialized equipment other than a smartphone and/or a specific wearable inertial sensor. To perform it, either of these two devices was placed on the dorsum of the hand to record the tremor in two arm positions (resting and posture) at a defined frequency rate for 30 s. The recorded data were analyzed using statistical methods, and it was found that some biomarkers in the frequency spectrum can contribute to differentiating physiological and pathological tremors and, in turn, differentiate PD from ET. The results allowed us to use these biomarkers, also called kinematic features, to train Machine Learning (ML) models to classify hand tremors accurately. Thus, a methodology was initially developed to train ML models in Matlab using the kinematic features of the linear acceleration [13] and angular velocity [14] signals of the hand tremor of 51 subjects previously recorded at the Hospital Clìnic de Barcelona with an iPhone 5 at a sampling frequency of 100 Hz [12]. For this work, the same methodology was implemented in Python and the Scikit-Learn machine learning library to train and test new ML models using the data from the 51 subjects and new data from 25 subjects recorded at the Centro de Trastornos del Movimiento (CETRAM) in Chile between November 2021 and January 2022. The new data were recorded at a sampling rate of 120 Hz using a wireless inertial sensor, Xsens DOT. The new data were sub-sampled at 100 Hz to homogenize the dataset due to the difference in the sampling frequency used for the two devices. The biomechanical analysis was performed in the frequency domain in the range of 3 to 10 Hz to calculate the kinematic features, as described in [13] and [14]. Finally, the sample was split in a 70–30 ratio; 70% for training and 30% for testing. The ML models implemented in the App showed an accuracy of 96.77% for differentiating between physiological tremors and 94.73% for differentiating pathological tremors (PD and ET). Finally, as a result of these three works, it was proposed to develop TremorSoft. This Android-based mobile application uses the built-in inertial sensors of the smartphone or the external wirelessly connected inertial sensor (Xsens DOT) to serve as a tool to support the differential diagnosis of PD and ET during routine clinical practice.

TremorSoft is a novel e-health application in the field of movement disorders research, whose main contribution lies in implementing ML algorithms that have a high efficiency to easily and quickly classify PD and ET. It is expected that this application can be used as an additional tool in the medical evaluation of patients with high suspicion of PD or ET that, through an alternative and non-invasive test procedure, allows the physician to have timely and reliable information to make a correct diagnosis of the patient. Especially for developing countries, where hospitals often have only simple tools and techniques at their disposal, a standardized, convenient and accurate low-cost tool to differentiate PD from ET would be of considerable help.

## 2. Software description

### 2.1. Software architecture

TremorSoft implements the 6-axis inertial sensor from the smartphone or a wearable sensor connected via Bluetooth to record and analyze hand tremor data, classifying them accordingly. The tool consists of two parts: (1) a smartphone application to record the demographic, clinical, and kinematic data of the subject to be evaluated, and (2) a web server where the recorded kinematic data are processed to classify them deferentially, accordingly, between HS, ET and PD by applying ML models to the features extracted from these data. The mobile application is built on the Android operating system and represents the front-end of the tool, written in Kotlin using the Android SDK and the Xsens DOT SDK. The target version of the operating system is Android 8.0 Oreo or higher. All processes running on the web server, the back-end, are hosted on the Heroku platform, written in Python. The Retrofit network library is implemented as a REST client to load and retrieve data from the back-end. Regarding the Authentication services, the Firebase platform is used. Fig. 1 shows the general architecture of the developed software.

### 2.2. Software functionality

#### 2.2.1. Front-end

The Front-end consists of an Android application in which users, physicians, and movement disorder specialists can record information about patients with suspected or diagnosed PD or ET. The information that can be recorded corresponds to primary clinical data, hand tremor signals, and, in the case of diagnosed patients, information related to their diagnosis and the treatment received. The mobile app guides the user to record hand tremor signals using the built-in inertial sensors (gyroscope and accelerometer) of the smartphone or an external inertial sensor (Xsens DOT) connected to it wirelessly. The recorded signals correspond to two positions, rest and posture, which are stored separately in two ArrayList class variables, *restData* and *postureData*. When the external sensor is used to record hand tremor

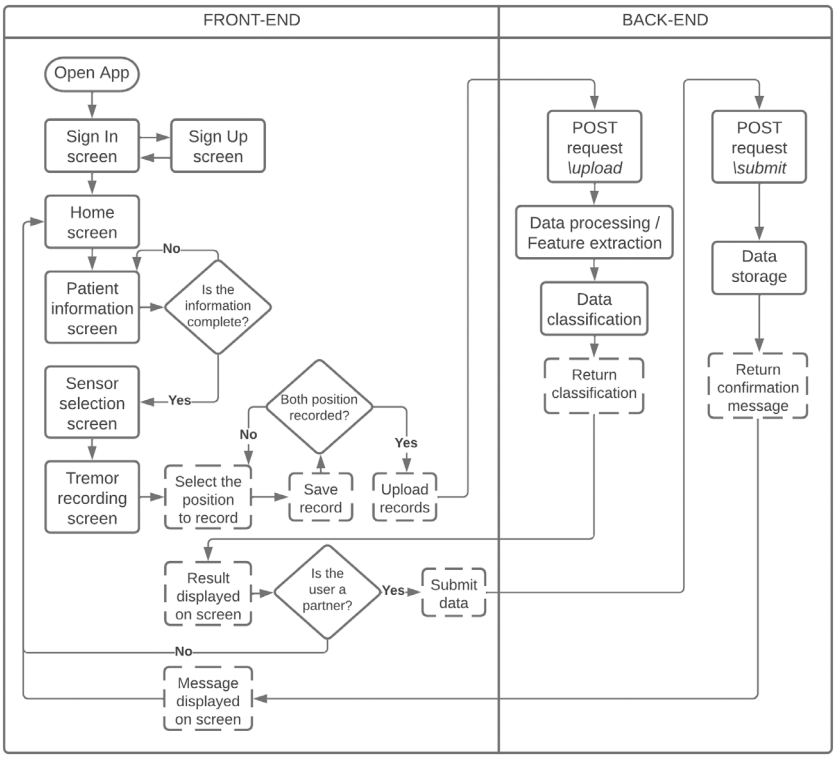



**Fig. 1.** TremorSoft flowchart.

signals, these are stored in the internal device memory, so they must first be exported to the application and then assigned to *restData* and *postureData* as appropriate. Once both positions are recorded and saved, the two lists are combined into a single list called *tremorData*. Then, a new variable of type JSONArray, called *dataArray*, is created from the latter list. This JSONArray is the one that is sent to the server hosted on the Heroku platform, where the signals are processed, analyzed, and classified. The application then receives and displays a message corresponding to the result returned by the server. The three possible messages the application can display according to the classification obtained are: (1) A pathological tremor has been identified. It has been classified as PD; (2) A pathological tremor has been identified. It has been classified as ET, or (3) The recorded tremor has been classified as a physiological tremor.

If the user who enters the application is an associated user, i.e., a user accredited as a TremorSoft collaborator, the application enables the *SEND* button. The user confirms the classification received by pressing this button and authorizes the submission and storage of the recorded data in the webserver database. When this process is initiated, the application takes each of the collected clinical and diagnostic data and stores them in a new JSONObject, called *patientDataJSON*, which is subsequently added to the JSONArray called *tremorsoftData*, a copy of *dataArray* which contains the recorded signals at rest and posture. Then, *tremorsofData* is sent via a POST request under the path */submit*. Finally, the server returns a confirmation message displayed on the application screen upon completing this task. Here the user can decide whether to restart the classification process on the same patient (*RESET* button) or perform a new test on a different patient (*NEW* button). Fig. 2 shows the user interface of the software.

### 2.2.2. Back-end

The back-end was developed as a RESTful API using Python and Flask and is deployed on the Heroku platform. The back-end has hosted all the functions and elements essential for processing, analyzing, and classifying the hand tremor records and storing these records and other patient data in the SQL database. The three main components of the TremorSoft web server are described below:

1. **Data preprocessing and extraction of kinematic features:** The server receives the JSONArray via the */upload* path from the mobile application and transforms it into a Pandas Dataframe for further processing. The signals are initially filtered during preprocessing to reduce sensor drifts and distortions due to various physical phenomena. Next, the Power Spectral Density (PSD) of each accelerometer and gyroscope axes is calculated. From the PSD of the components, kinematic features are extracted and then evaluated by ML models.
2. **Classification of hand tremor using ML:** The server hosts the classification models developed based on the methodology and results obtained in previous studies [12–14]. The Classification function uses a model and specific kinematic features, previously extracted with the Preprocessing function, to initially classify the recorded tremor as physiological or pathological. If the tremor is classified as pathological by the first model, a second model is used to classify the tremor of the subject as PD or ET. Although many kinematic features have been extracted in previous works, the classification models only use a small number of these, i.e., only those that, together with the model, provide high predictive power.
3. **Storing patient data with a confirmed diagnosis:** An SQL database was linked to the server using the Heroku Postgres service for data upload and storage. Subject data is uploaded to the web server in a POST request via the */upload* path once it is verified that all required information has been supplied. As shown in Figs. 3 and 4, the data set that is stored for each patient contains 25

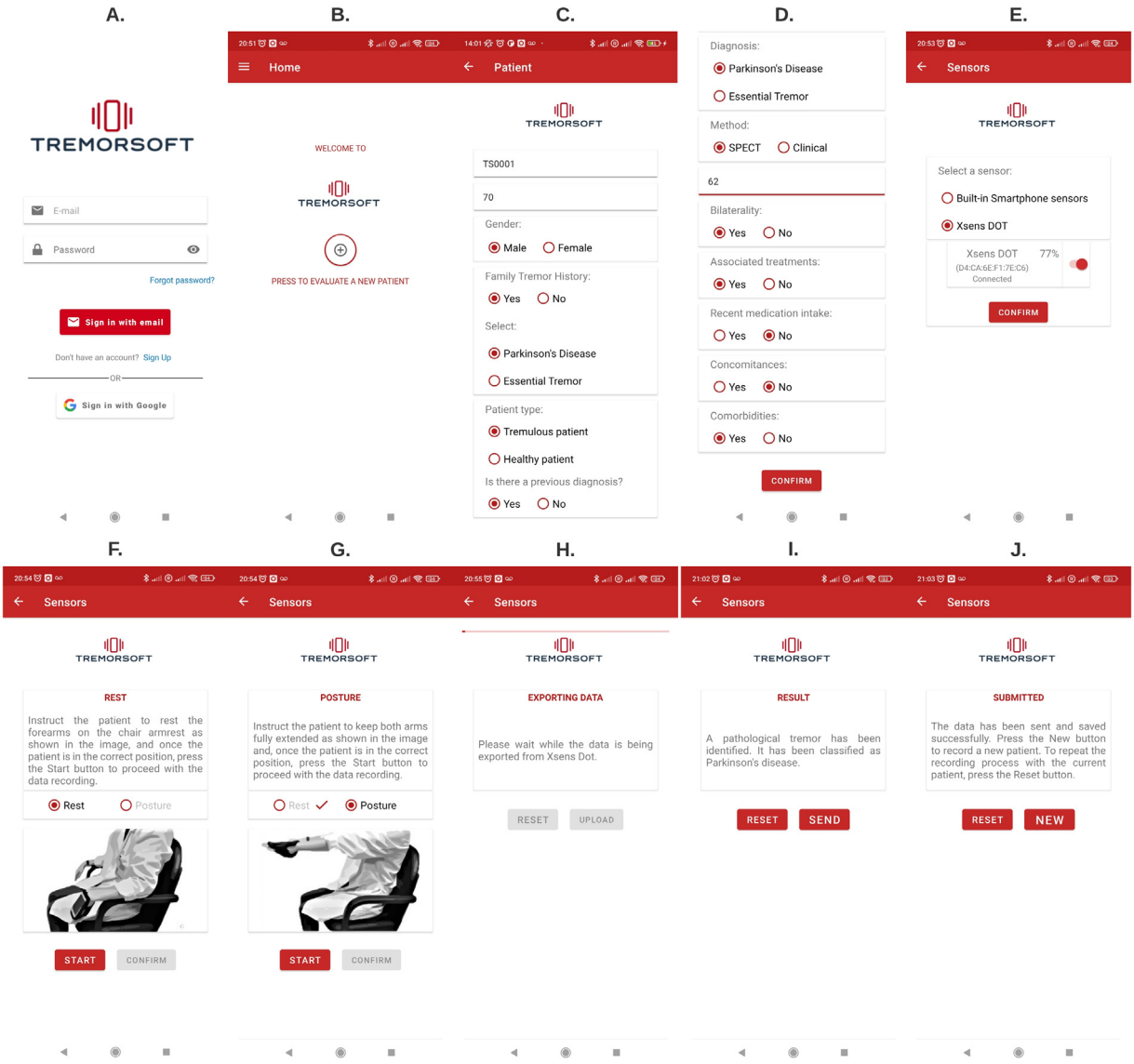



**Fig. 2.** Overview of the TremorSoft user interface dialog screens: (A) Login screen; (B) Home screen; (C) Basic patient data; (D) Patient diagnostic data; (E) Sensor selection for recording hand tremor; (F) Recording hand tremor in resting position; (G) Recording hand tremor in postural position; (H) Export of records when using Xsens DOT; (I) Tremor classification result returned by the web server; (J) Confirmation of data submission and saving to the web server.

columns: Patient identification (Patient_id), Age, Gender, Family history of tremor (F_history), Classification received by ML models (Classification), Previous diagnosis of the patient (Diagnosis), Method of diagnosis (Method), Age of onset or diagnosis of the disease (O_age), Bilaterality, Treatments received (Treatments), Medication intake prior to registration (Medication), Concomitances, Comorbidities, Accelerometer signals at Rest (AccX_r, AccY_r, AccZ_r), Accelerometer signals at Posture (AccX_p, AccY_p, AccZ_p), Gyroscope signals at Rest (GyrX_r, GyrY_r, GyrZ_r) and Gyroscope signals at Posture (GyrX_p, GyrY_p, GyrZ_p).

## 3. Impact

The medical relevance of this e-Health App is framed in the achievement of new advances and knowledge beyond the state of the art of movement disorder testing methods, as this tool will serve as an additional evaluation technique to help differentiate pathological tremors that, in some cases, mainly in the early stages of diseases, are not easy to identify. Furthermore, compared to the one obtained with SPECT, the level of reliability that this app will achieve will be high enough to help neurologists correctly evaluate and identify movement disorders and, in turn, measure their severity. Likewise, the knowledge generated from this tool will represent a significant scientific contribution to improving the differential diagnosis of different movement disorders.

This application aims to provide more objective information to facilitate decision-making and, above all, reduce waiting times before receiving a final diagnosis, making it possible for patients to access appropriate treatment promptly. Three stakeholders will benefit from the success of the application. First, the attending physician can use the app to support the initial evaluation of the disease, especially in undiagnosed, early, or complicated cases. For the patient, correct and early treatment can positively affect the health condition, helping reduce complications and prolong the quality of life. Finally, for healthcare systems, the impact is mainly financial where. For example, in Spain, medical costs per patient with PD can amount to 17,000 € per year [15].

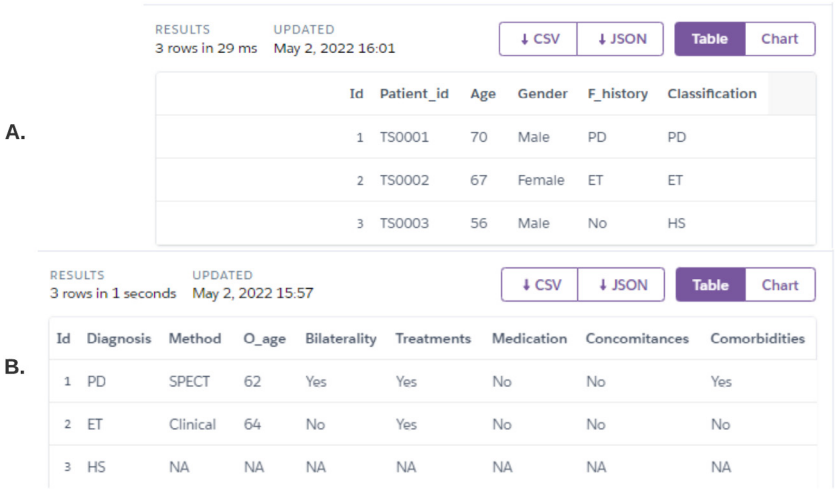



**Fig. 3.** Patient data stored in the SQL database hosted on the Heroku server: A. Basic patient data and tremor classification; B. Diagnosis-related data.

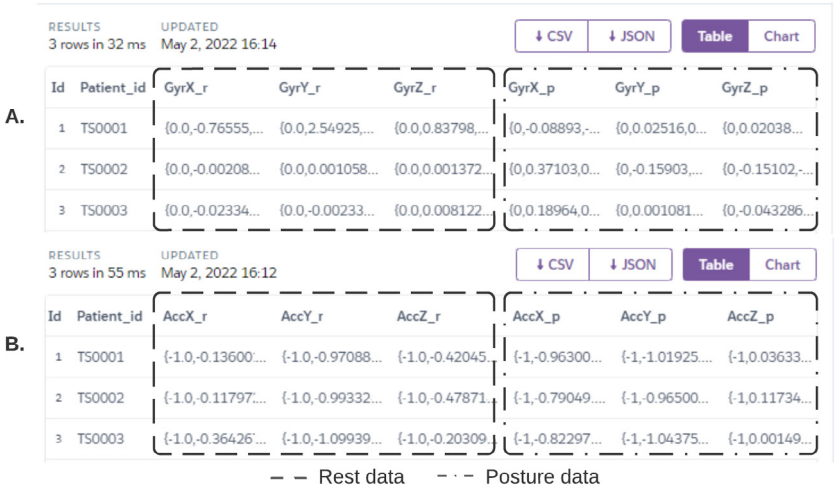



**Fig. 4.** Data from the resting and postural tremor records stored in the SQL database hosted on the Heroku server: (A) Records of angular velocities from the gyroscope; (B) Records of linear accelerations from the accelerometer.

The use of this tool could contribute to the reduction of expenses incurred on erroneous or ineffective treatments and even avoid the need to use expensive techniques or technologies. It is important to highlight that the classification given by the App cannot be considered as a definitive diagnosis, it would serve as an added value in the decision-making of physicians and movement disorder specialists. They are the ones who would execute these assessments and not the patients, taking into account other clinical criteria of the patient besides the hand tremor. Likewise, given the short time it takes to register and classify the patient, the treating physician can repeat the test if it is considered that a false positive or false negative has occurred due to an error at the time of registering the patient.

In future work, we intend to seek the collaboration of different movement disorders centers to expand the database with records of patients with confirmed diagnoses in order to perform constant improvement and maintenance of the implemented models and, thus, to have a higher degree of reliability in the classification made by the models. Taking into account the nature of the data that may be collected from both the user (Name, email, profile picture, etc.) and the patients (Age, gender, diagnosis, etc.), we will ensure that the final version of TremorSoft complies with all standards and measures imposed by the General Data Protection Regulation (GDPR) by performing the encryption of personal data, preventing unauthorized access to this data and constantly evaluating the security measures implemented. Finally, we also plan to add a new function in the mobile application that will allow sending reports to the email of physicians and specialists who use the application in their clinical routine.

## 4. Conclusions

This work provides a quantitative, easy-to-use, non-invasive, and cost-effective method that can be used as a supportive tool in diagnosing PD and ET based on recording the hand tremor. The tremor classification result is available in a short time during the medical evaluation by the physician, either in person or remotely. The combination of clinical information with kinematic feature information for ML model training is the key to the functionality of this tool, providing the application with increased classification accuracy. Typically, the classification of these motor disorders focuses on obtaining one or more kinematic biomarkers; however, the heterogeneity of both diseases makes this approach difficult, and we believe that complementing clinical data with kinematic biomarkers is more efficient.

## Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Antonio J. Sánchez Egea reports financial support was provided by Polytechnic University of Catalonia. Antonio J. Sánchez Egea reports a relationship with Polytechnic University of Catalonia that includes: employment. Julián D. Loaiza Duque, Antonio J. Sánchez Egea and Hernán A. Gónzalez Rojas have patent #2206021281741 licensed to Safe Creative.

## Data availability

No data was used for the research described in the article

## Acknowledgments

This work was supported by Serra Húnter Programme (Generalitat de Catalunya).

## Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.softx.2023.101393.

## References

[1] Woods AM, Nowostawski M, Franz EA, Purvis M. Parkinson's disease and essential tremor classification on mobile device. Pervasive Mob Comput 2014;13:1–12. http://dx.doi.org/10.1016/j.pmcj.2013.10.002, https://linkinghub.elsevier.com/retrieve/pii/S1574119213001260.

[2] Algarni M, Fasano A. The overlap between essential tremor and Parkinson disease. Parkinsonism Related Disorders 2018;46:S101–4. http://dx.doi.org/10.1016/j.parkreldis.2017.07.006, https://linkinghub.elsevier.com/retrieve/pii/S1353802017302390.

[3] di Biase L, Brittain J-S, Shah SA, Pedrosa DJ, Cagnan H, Mathy A, et al. Tremor stability index: A new tool for differential diagnosis in tremor syndromes. Brain 2017;140(7):1977–86. http://dx.doi.org/10.1093/brain/awx104, https://academic.oup.com/brain/article/140/7/1977/3782569, http://www.ncbi.nlm.nih.gov/pubmed/28459950, http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC5493195.

[4] Hristova AH, Koller WC. Early Parkinson's disease: What is the best approach to treatment. Drugs Aging 2000;17(3):165–81. http://dx.doi.org/10.2165/00002512-200017030-00002, URL http://www.ncbi.nlm.nih.gov/pubmed/11043817.

[5] Tarakad A, Jankovic J. Essential Tremor and Parkinson's Disease: Exploring the Relationship. Tremor Other Hyperkinetic Mov 2019;8:589. http://dx.doi.org/10.5334/tohm.441, URL https://tremorjournal.org/article/10.5334/tohm.441/.

[6] Benamer HTS, Patterson J, Grosset DG, Booij J, de Bruin K, van Royen E, et al. Accurate differentiation of Parkinsonism and essential tremor using visual assessment of [123I]-FP-CIT SPECT imaging: The [123I]-FP-CIT study group. Mov Disorders 2000;15(3):503–10. http://dx.doi.org/10.1002/1531-8257(200005)15:3<503::AID-MDS1013>3.0.CO;2-V.

[7] Darcourt J, Booij J, Tatsch K, Varrone A, Vander Borght T, Kapucu OL, et al. EANM procedure guidelines for brain neurotransmission SPECT using (123)I-labelled dopamine transporter ligands, version 2. Eur J Nucl Med Mol Imaging 2010;37(2):443–50. http://dx.doi.org/10.1007/s00259-009-1267-x, URL http://www.ncbi.nlm.nih.gov/pubmed/19838702.

[8] Song I-U, Park J-W, Chung S-W, Chung Y-A. Brain SPECT can differentiate between essential tremor and early-stage tremor-dominant Parkinson's disease. J Clin Neurosci 2014;21(9):1533–7. http://dx.doi.org/10.1016/j.jocn.2013.11.035, https://pubmed.ncbi.nlm.nih.gov/24814853/, https://linkinghub.elsevier.com/retrieve/pii/S0967586814000940, http://www.ncbi.nlm.nih.gov/pubmed/24814853.

[9] Orozco-Arroyave JR, Vásquez-Correa JC, Klumpp P, Pérez-Toro PA, Escobar-Grisales D, Roth N, et al. Apkinson: the smartphone application for telemonitoring Parkinson's patients through speech, gait and hands movement. Neurodegenerative Dis Manag 2020;10(3):137–57. http://dx.doi.org/10.2217/nmt-2019-0037, URL http://www.ncbi.nlm.nih.gov/pubmed/32571150.

[10] Kubben PL, Kuijf ML, Ackermans LP, Leentjes AF, Temel Y. TREMOR12: An open-source mobile app for tremor quantification. Stereotact Funct Neurosurg 2016;94(3):182–6. http://dx.doi.org/10.1159/000446610, https://www.karger.com/Article/FullText/446610, http://www.ncbi.nlm.nih.gov/pubmed/27395052.

[11] Linares-del Rey M, Vela-Desojo L, Cano-de la Cuerda R. Mobile phone applications in Parkinson's disease: A systematic review. Neurología (Engl Ed) 2019;34(1):38–54. http://dx.doi.org/10.1016/j.nrleng.2018.12.002, URL https://linkinghub.elsevier.com/retrieve/pii/S2173580818301974.

[12] Barrantes S, Sánchez Egea AJ, González Rojas HA, Martí MJ, Compta Y, Valldeoriola F, et al. Differential diagnosis between Parkinson's disease and essential tremor using the smartphone's accelerometer. In: Chen R, editor. PLoS One 2017;12(8):e0183843. http://dx.doi.org/10.1371/journal.pone.0183843, URL https://dx.plos.org/10.1371/journal.pone.0183843.

[13] Loaiza Duque JD, González-Vargas AM, Sánchez Egea AJ, González Rojas HA. Using machine learning and accelerometry data for differential diagnosis of Parkinson's disease and essential tremor. In: Communications in computer and information science, vol. 1052. Springer; 2019, p. 368–78. http://dx.doi.org/10.1007/978-3-030-31019-6{_}32, URL http://link.springer.com/10.1007/978-3-030-31019-6_32.

[14] Duque JDL, Egea AJS, Reeb T, Rojas HAG, Gonzalez-Vargas AM. Angular Velocity Analysis Boosted by Machine Learning for Helping in the Differential Diagnosis of Parkinson's Disease and Essential Tremor. IEEE Access 2020;8:88866–75. http://dx.doi.org/10.1109/ACCESS.2020.2993647, URL https://ieeexplore.ieee.org/document/9091038/.

[15] García-Ramos R, López Valdés E, Ballesteros L, Jesús S, Mir P. Informe de la Fundación del Cerebro sobre el impacto social de la enfermedad de Parkinson en España. Neurología 2016;31(6):401–13. http://dx.doi.org/10.1016/j.nrl.2013.04.008, URL https://linkinghub.elsevier.com/retrieve/pii/S0213485313001114.

# Appendix G

# Safe Creative license: *Certificate of Declarative Inscriptions of Rights: TremorSoft App*

safecreative

# CERTIFICATE OF INSCRIPTION

OF THE WORK

# TREMORSOFT_APP

## Technological proof

Made on **June 2, 2022** at **9:32 AM UTC**.
At the request of: **Universitat Politècnica de Catalunya** (User ID: 2107203916096) ("the holder").
Which proves the possession of the digital file on the indicated date, for the purposes of protection and defense of intellectual property rights on the principle "Prior tempore, potior iure".

Proof id 2206021281741.

Consisting of the following digital files:

File provided by "the holder".

**File name:** tremorsoft.apk
**File size:** 15269015 bytes
**SHA1 hash:** 3dec1fec4d45a166073beed8deec94294807c812
**SHA256 hash:** 832cfe8acf0da217c1177c860e41627e04adbfc3d9cbdb61ec2bb79f3adbfe5e
**SHA512 hash:** d5c14d8641e72025d414d80ca4e322ea9d6714b1ce067d796204914cae0985317ede7518fcccf52127b2d5ce7808c48f34f5b2c3063440ae67da99712df1feb5

Identification file[1]: contains the hashes of the file provided .

**File name:** 2206021281741.txt

Guarantee file[1]: time stamps of the identification file.

**Timestamp provided by Safe Creative:** timestamp-safecreative.asn
**Authority identity:**
C=ES,ST=Zaragoza,L=Zaragoza,O=Safe Creative,OU=Safe Creative,CN=Time Stamp Authority Server,E=pki@safecreative.com
**Timestamp provided by authorized third-party:** timestamp-external.asn
**Authority identity:**
C=ES,O=Firmaprofesional SA,organizationIdentifier=VATES-A62634068, CN=FIRMAPROFESIONAL CLOUD QUALIFIED TSU - 2021

[1] Files embedded in this pdf document.

CERTIFICATE OF DECLARATIVE INSCRIPTIONS OF RIGHTS

# TREMORSOFT_APP

IDENTIFIER: 2206021281741

Safe Creative, on Jun 2, 2022 at 9:48 AM UTC, hereby certifies the registration of the following information in the electronic copyright registry.

## Proof of the work:

Carried out on Jun 2, 2022 9:32 AM UTC at the request of: Universitat Politècnica de Catalunya ("the holder").

Which proves the possession of the digital file provided by the holder on the indicated date, for the purposes of protection and defense of intellectual property rights on the principle "Prior tempore, potior iure".

## Declarative inscriptions of rights:

Holder: **Universitat Politècnica de Catalunya** Rights of: **Titularidad** Date-time: **Jun 2, 2022 9:32 AM** % **100.00**

On the date and time of issuing this informative file, the rights associated to this work according to the registry are described under "All rights reserved".

# Appendix H

# Short Research Stay: *Facultad de Ingeniería Mecánica - Universidad Tecnológica de Pereira (UTP)*

**Note from:**

**Servei de Biblioteques, Publicacions i Arxius de la UPC**

Appendix H removed for data protection purposes

# Appendix I

# Collaboration Agreement: *Associació Catalana per al Parkinson*

**Note from:**

**Servei de Biblioteques, Publicacions i Arxius de la UPC**

Appendix I removed for data protection purposes