Final Master Work

Master in Data Science and Computer Engineering

Development of an intelligent mHealth system to predict trunk endurance and muscle fatigue

Author
Ignacio Díaz Reyes

Directed by
Dr. D. Miguel Damas Hermoso
Dr. D. Oresti Baños Legran

Higher Technical School of Information Technology and Telecommunications Engineering

—

Granada, September/2016

Development of an intelligent mHealth system to predict trunk endurance and muscle fatigue

Ignacio Díaz Reyes

Palabras clave: Salud móvil, salud digital; tests resistencia; fatiga muscular; minería de datos; electromiograf'ia; sensores wearables

Resumen

El dolor lumbar continua siendo una de las principales causas de absentismo laboral en el mundo. Además del impacto socieconómico que genera, la edad a la que comienzan los primeros síntomas es cada vez menor. En consecuencia, el número de especialistas que dedican su trabajo a realizar planes de prevención para la zona lumbar es mayor. Por otro lado, el contínuo crecimiento del mercado de sensores vestibles permite a los expertos obtener un preciso feedback de las mejoras de sus pacientes de forma casi diaria. Debo a esto, el pasado aõ se presentó mDurance, un nuevo sistema de salud móvil capaz de dar apoyo a los especialistas en la evaluación funcional de la resistencia y la actividad muscular del tronco.

Debido a su buena acogida, en este trabajo se desea continuar con el desarrollo y la verificación de la utilidad del sistema. En este trabajo se presenta un nuevo sistema de almacenamiento en la nube, formado por un servicio de back-end y una aplicacin API Rest que pretenden ser el principal motor de datos para poder aplicar técnicas de minería de datos en futuras aplicaciones y estudios. Además, para verificar la utilidad del sistema, se han realizado pruebas a partir del nuevo sistema desarrollado en un equipo profesional de fútbol. Este estudio pretende ser el punto de inicio para la elaboración de un nuevo sistema de clasificación de la actividad muscular en la zona lumbar, a partir de aplicar las conocidas técnicas de clustering y minería de datos en los nuevos datos generados por la aplicación.

Development of an intelligent mHealth system to predict trunk endurance and muscle fatigue

Ignacio Díaz Reyes

Keywords: mHealth; digital health; endurance tests; muscle fatigue; data mining; electromyography; wearable sensors

Abstract

Low back pain remains a major cause of absenteeism in the world. In addition to the socio-economic impact generated by this cause, the age at which the first symptoms appear is decreasing. Consequently, the number of specialists who dedicate their work to make prevention plans for the lumbar area is greater. On the other hand, the continued market growth wearable sensors allows experts to obtain a precise feedback from improvements in his patients almost daily basis. Due to this, last year mDurance was presented, a novel mobile health system aimed at supporting specialists in the functional assessment of trunk endurance and muscle activity by using wearable and mobile devices.

Because of its popularity, this work wants to continue with the development and verification of the usefulness of the system. This works presents a new storage system in the cloud, formed by a back-end service and API Rest application. They hope to be the main data engine to can apply data mining techniques and applications in future studies. In addition, to verify the usefulness of the system, the tests have been tested by the new system developed into a professional football team. This study aims to be the starting point for the development of a new classification system of muscle activity in the lower back, from applying the known techniques of clustering and data mining in the new data generated by the application.

Yo, **Ignacio Díaz** Reyes, alumno de la titulación Máster Universitario Oficial en Ciencia de Datos e Ingeniería de Computadores de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, con DNI 76658187S, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Máster en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Fdo: Ignacio Díaz Reyes

Granada a 15 de Septiembre de 2015.

Dr. D. Miguel Damas Hermoso, Profesor del Departamento de Arquitectura de los Computadores de la Universidad de Granada.

Informa:

Que el presente trabajo, titulado ***Development of an intelligent mHealth system to predict trunk endurance and muscle fatigue***, ha sido realizado bajo su supervisión por **Ignacio Díaz**, y autorizo la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expide y firma el presente informe en Granada a 12 de Septiembre de 2016.

El director:

Dr. D. Miguel Damas Hermoso

Acknowledgments

> Sin continuo crecimiento y progreso, tales palabras como mejora, logro y éxito no tienen significado. Benjamin Franklin.

Quiero comenzar agradeciéndole a mi madre haber sabido transmitirme siempre su carácter emprendedor y luchador, ya que sin ese contínuo esfuerzo de superación que siempre he visto en ella esto no hubiese sido posible. Por supuesto a mi padre, cuya figura siempre aparece en mi mente en los momentos en los que más ánimo necesito. En general a mi familia que me han apoyado en todo momento.

Otra persona imprescindible es Soledad Ortíz, que ha sido la persona con la que he compartido todos los buenos momentos y otros no tan buenos, pero siempre ha estado ahí.

No me olvido de todos mis amigos que han tenido que escuchar hasta el más mínimo detalle del trabajo aportando su granito de arena.

Finalmente, agradecerles a Miguel Damas y a Oresti Baños su apoyo, su tiempo y su dedicación en la realización de este trabajo.

Muchas gracias todos.

Contents

List of Figures

List of Tables

1

Introduction

1.1 Motivation

Low back pain remains a major cause of absenteeism in the world. In addition, it continues to decrease the age at which the first signs of fatigue or discomfort in the low back begin to appear. For this, they are increasingly experts are beginning to make plans to improve prevention specialized both the endurance and the discharge of these important muscles in daily activity. In addition, the continued market growth of wearable devices allows experts to obtain precise feedback from improved patients. Due to this, last year mDurance was presented, a novel mobile health system aimed at supporting specialists in the functional assessment of trunk endurance and muscle activity by using wearable and mobile devices (10, 11). mDurance is based on the use of two wearable sensors, one inertial (IMU) and other electromyography (EMG) used to monitor the posture and the muscle activity respectively. All this is managed by a mobile application that allows the specialist to get an online feedback about the patient outcomes. It also allows them to store the historical of each patient and to check their progress on the results achieved by them during all the sessions. Due to its good reception, it is proposed to continue with its development and to study its efficacy in a real study. In addition, we will test the most knowledge acquired during the master of science data and engineering computer and we apply clustering and data mining techniques to extract the major number of conclusions about the trunk endurance and muscle activity.

1.2 Context

This section presents the main context in which is placed this work. This section is divided into three parts: the low back pain context 1.2.1, the data mining application context 1.2.2 and the wearable sensors apply to a mobile health context 1.2.3.

1.2.1 Low back pain

Actually, low back pain (LBP) continues to be considered an extremely common health problem, and the major cause of activity limitation and work absence in the world. LBP is ahead of 290 other conditions and causing an enormous economic burden on individuals, families, communities, industry and governments. Furthermore, LBP is ranked sixth in terms of overall burden of disability-adjusted life years (DALYs). DALYs is defined as the sum of two components: years of life lost due to premature mortality (YLLs) and years lived with disability (YLDs). DALYs is an absolute measure of health loss; they count how many years of healthy life are lost due to death and non-fatal illness or impairment. They reflect the number of individuals who are ill or die in each age-sex group and location. Population size and composition influences the number of DALYs in a population. This condition increased from 46.06 million (95% uncertainly intervals: 31.6-63.5 million) in 1990 to 72.317 million (95% UI: 49.05-99.8 million) in 2013. To help in the rapid dissemination of results, it is going to be used a visualization tool developed by the Institute for Health Metrics and Evaluation (IHME) (1), where it is possible to consult all kind of statistics about LBP and other conditions. For example, Figure 1.1 displays the global low back pain number for both sexes and all ages globally with respect to DALYs in 2013 in comparison with 1990.

In addition, in the Figure 1.2 anyone can also compare the top 10 causes of disability from 2010 to 2013 in the world and in Spain. Low back pain disease remains the leading cause most YLDs in the world. YLDs increased 6.18% compared to 2010. The ranking shows that low back pain was the second cause of disability in Spain in 2013. Despite dropping one position in the ranking, the number of YLDs increased by 4.95% caused by this disease. With respect to regional statistics, it is also important to display the number of YLDs that LBP

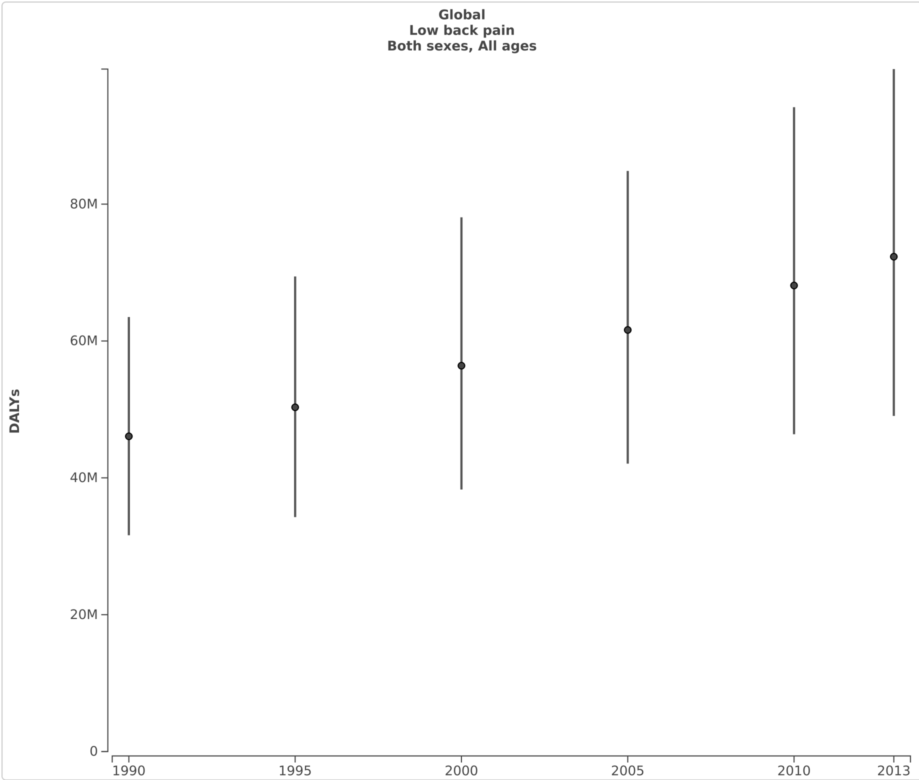


Figure 1.1: Low back pain number for both sexes and all ages globally with respect to DALYs in 2013 in comparison with 1990. Figure reprinted from (1)

causes by world region, such as it is shown in Figure 1.3. In this case, Asia leads the ranking. Concretely, China leads the ranking North and Central Europe and North American remain the leaders in the same way with respect to the YLDs.

Figure 1.4 displays the evolution of LBP in 2013 function of people aged in both sexes versus the same evolution in 2010. It is important to note that the biggest percentages are from 20 to 50 years. It means that the most of cases occur during the youth and the laboral life. If they are compared the three segments most significant, it can be demonstrated that in 3 years, the LBP was increased by 0.10% of average. Thus, it confirms that LBP continues increasing each year.

A recent study (12) shows that in 2012, LBP reached a point prevalence of 11.9%, one-month prevalence of 23.2% , one-year prevalence was of 38% and the lifetime prevalence was 39.9%. In chronic LBP has been estimated that the average age-related prevalence is 15% in adults and 27% in the elderly. Furthermore,

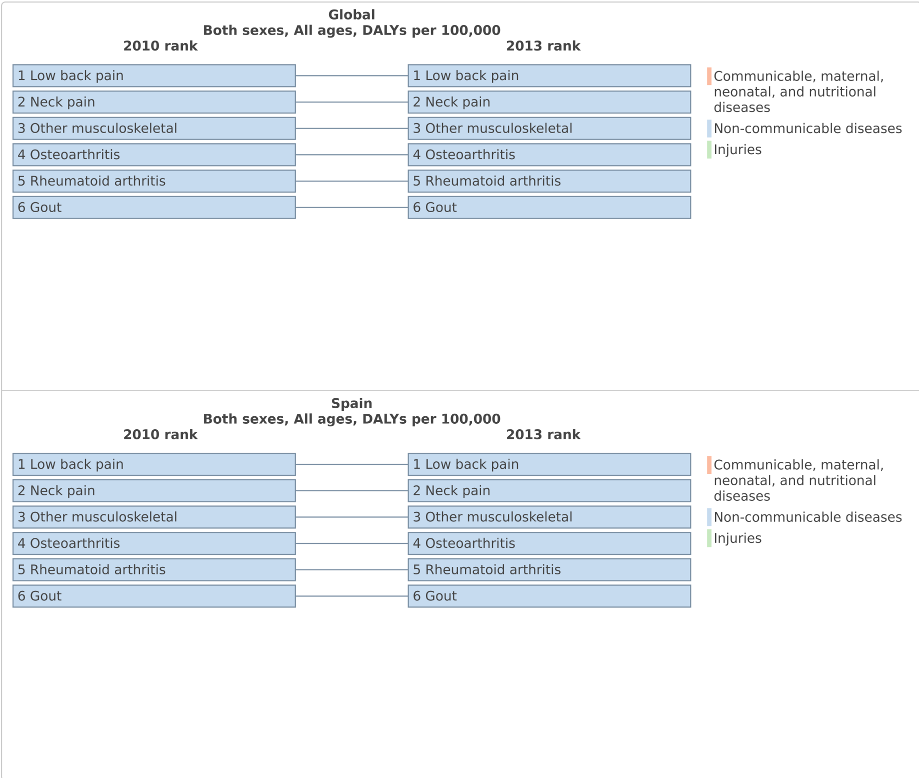


Figure 1.2: Ranking top musculoskeletal causes of disability in 2010 and 2013. Figure reprinted from (1)

LBP had the highest prevalence among women aged between 40 and 80 years. In addition, LBP has an enormous social and economic impact (13), and is a leading cause of absenteeism in all professions (14).

1.2.2 Data mining

Recent years have witnessed a seamless increase in the ability to collect data from various sensors, devices, in different formats, from independent or connected applications. This data flood has outpaced our capability to process, analyze, store and understand these datasets. Furthermore, with mobile phones becoming the sensory gateway to get realtime data on people from different aspects, the vast amount of data that mobile carriers could improve our daily life potentially. In addition, it can be foreseen that Internet of things (IoT) applications will raise

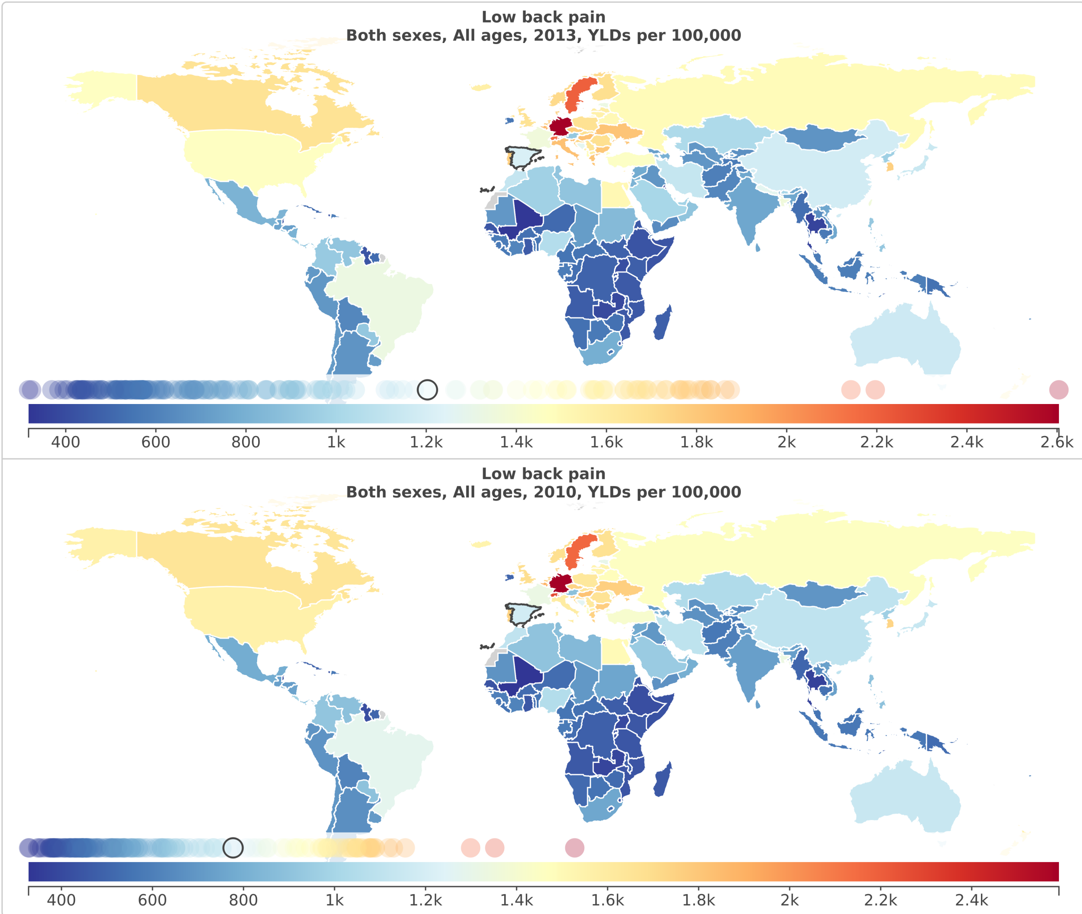


Figure 1.3: Global YLDs of LBP. Figures obtained from (1).

the scale of data to an unprecedented level (15). Data mining can be defined as the process of finding previously unknown patterns and trends in databases and using that information to build predictive models. It aims to identify valid, novel, potentially useful, and understandable correlations and patterns in data that allow to detect patterns which can be difficult to detect for humans (16). Data mining is not new and it has been widely used by financial institutions, for credit scoring and fraud detection; marketers, for direct marketing and cross-selling or up-selling; retailers, for market segmentation and store layout; and manufacturers, for quality control and maintenance scheduling. In healthcare, data mining is becoming increasingly popular. Several factors have motivated the use of data mining applications in healthcare:

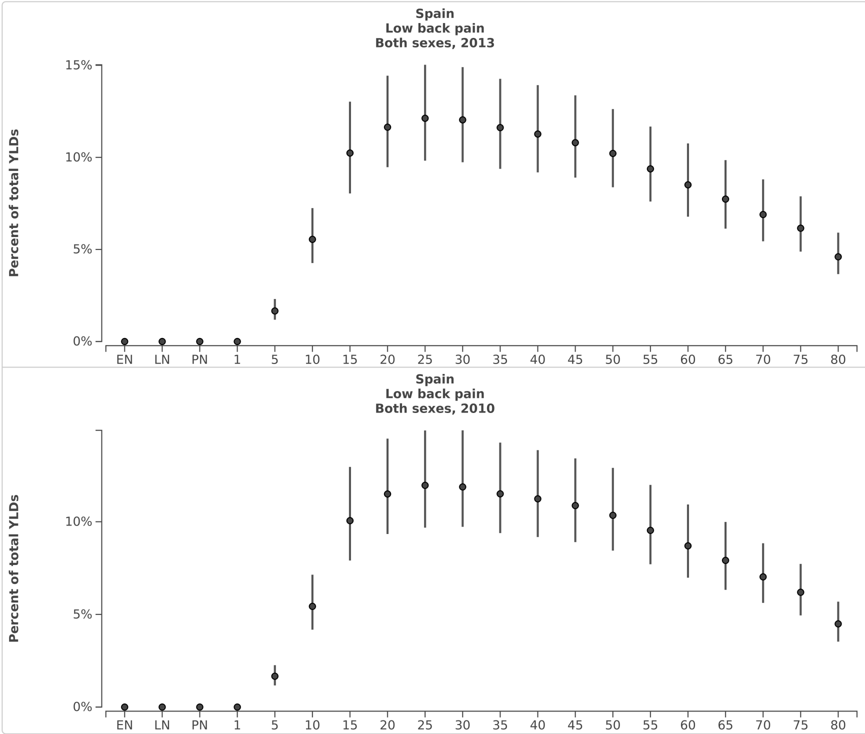


Figure 1.4: Evolution of LBP in percent of total YLDs for 2013 in comparison with 2010 in function of all ages. Figures reprinted from (1)

- The existence of medical insurance fraud and abuse, for example, has led many healthcare insurers to attempt to reduce their losses by using data mining tools to help them find and track offenders (17). Fraud detection using data mining applications is prevalent in the commercial world, for example, in the detection of fraudulent credit card transactions. Recently, there have been reports of successful data mining applications in healthcare fraud and abuse detection.

- Another factor is that the huge amounts of data generated by healthcare transactions are too complex and voluminous to be processed and analyzed by traditional methods. Data mining can improve decision-making by discovering patterns and trends in large amounts of complex data. Such analysis has become increasingly essential as financial pressures have heightened the need for healthcare organizations to make decisions based on the analysis of clinical and financial data. Insights gained from data mining can influence cost, revenue, and operating efficiency while maintaining a high level of care (18, 19).

- Healthcare organizations that perform data mining are better positioned to meet their long-term needs, (20) argue. Data can be a great asset to healthcare organizations, but they have to be first transformed into information.

- Yet another factor motivating the use of data mining applications in healthcare is the realization that data mining can generate information that is very useful to all parties involved in the healthcare industry. For example, data mining applications can help healthcare insurers detect fraud and abuse, and healthcare providers can gain assistance in making decisions, for example, in customer relationship management. Data mining applications also can benefit healthcare providers, such as hospitals, clinics and physicians, and patients (21, 22), for example, by identifying effective treatments and best practices. This idea will be developed in next chapters since it is one of main goals for this study.

There is vast potential for data mining applications in healthcare. Generally, these can be grouped as:

- The evaluation of treatment effectiveness. Data mining applications can be developed to evaluate the effectiveness of medical treatments. By comparing and contrasting causes, symptoms, and courses of treatments, data mining can deliver an analysis of which courses of action prove effective. For example, the outcomes of patient groups treated with different drug regimens for the same disease or condition can be compared to determine which treatments work best and are most cost-effective (23).

- Management of healthcare. To aid healthcare management, data mining applications can be developed to better identify and track chronic disease states and high-risk patients, design appropriate interventions, and reduce the number of hospital admissions and claims.

- Customer relationship management. While customer relationship management is a core approach in managing interactions between commercial organizationstypically banks and retailersand their customers, it is no less important in a healthcare context. Customer interactions may occur through call centers, physicians offices, billing departments, inpatient settings, and ambulatory care settings. As in the case of commercial organizations, data mining applications can be developed in the healthcare industry to determine the preferences, usage patterns, and current and future needs of individuals to improve their level of satisfaction. These applications also can be used to predict other products that a healthcare customer is likely to purchase, whether a patient is likely to comply with prescribed treatment or whether preventive care is likely to produce a significant reduction in future utilization.

- Detection of fraud and abuse. Data mining applications that attempt to detect fraud and abuse often establish norms and then identify unusual or abnormal patterns of claims by physicians, laboratories, clinics, or others. Among other things these applications can highlight inappropriate prescriptions or referrals and fraudulent insurance and medical claims (24).

Data mining is also used in the sports. In the world, a huge number of games are available where each and every day the national and international games are to be scheduled, where a huge number of datas are to be maintained. The data mining tools are applied to give the information as and when we required. In the sports world the vast amounts of statistics are collected for each player, team, game, and season. In the game sports the datas are available in the form of statistical form where data mining can be used and discover the patterns, these patterns are often used to predict the future forecast. Data mining can be used for scouting, prediction of performance, selection of players, coaching and training and for the strategy planning (25). In addition, the data mining is also used for improve the performance of the athletes. For example, there are progressively more professional teams using a myriad of wearable sensors and data mining techniques to extract accurate conclusions of the fitness of the players. According to (26), researches indicate that data mining can be used on physical aptitude test data in order to predict future physical performance. This represents an research opportunity, since there are a number of independent research organizations devoted to the increase of knowledge and understanding of their respective sport. These associations serve as central hubs for idea exchange and collaboration among sports experts and researchers. Many of them provide online databases and publish journals or newsletters. As the majority of these organizations are nonprofit, their work is not driven by monetary gain but rather by a pure passion for sport and its research (27).

Data mining applications can greatly benefit the healthcare industry. However, they are not without limitations. Healthcare data mining can be limited by the accessibility of data, because the raw inputs for data mining often exist in different settings and systems, such as administration, clinics, laboratories and more. Hence, the data have to be collected and integrated before data mining can be done. While several authors and researchers have suggested that a data warehouse be built before data mining is attempted, that can be a costly and time-consuming project (24).

1.2.3 Wearable sensors and mHealth

Within this context of clinical needs is emerging the digitization of medical procedures. During the last years, the use of devices and software in healthcare disciplines has become more common due to the constant technological improvement (14, 28, 29). Digital technologies can serve to cope with some of the limitations introduced by human errors during the practice of medical procedures. There are different factors attributable to the development of this type of systems:

- The globalization of health systems (30). A number of major actors involved in global health currently are engaged in health systems strengthening, including the World Health Organization, the World Bank, the G8, and several global health initiatives and private foundations. Many have come to pay attention to the issue only in the past few years.

- The demand by healthcare users for novel forms of treatment and prevention (31). Prehabilitation is a process that will help prepare to people by improving their functional capacity before encountering the event. This concept has been employed by athletes in an effort to reduce the risk of injuries and help accelerate their rebound from any injuries.

- The need of reduction of healthcare cost. Consistently, as many as 20% of patients who participate in shared decision making choose less invasive surgical options and more conservative treatment than do patients who are not consulted (32).

- The major advances in information and communication technologies (33).

Telehealth, eHealth, Social Health, and Health IT are some of the most prominent areas in which telecommunications and computer technologies are combined to expedite and enhance healthcare procedures. Currently, at the forefront of the digital health revolution is the so-called mobile health (mHealth) (34),which refers to the practice of medicine and public health supported by mobile devices and applications. mHealth technologies offer real-time monitoring and detection of changes in health status, support the adoption and maintenance of a healthy lifestyle, provide rapid diagnosis of health conditions, and facilitate the implementation of interventions ranging from promoting patient self-care to providing remote healthcare services. The interest in this domain has been boomed by the growth of wearables and mobile technologies(35). In regard the global wearable technology market, in Figure 1.5 is displayed the continuous growth in each one the application fields of wearable devices since 2011. Starting at less than 120 millions units in 2013, the total wearables market worlwide is expected to top 230 millions unit by 2019, nearly doubling in volume withun the six-year period. And unlike, some other markets, revenues in this segment are forecast to grow even faster than unit shipments, more than tripling in value to over 32$ billion by 2019, up from 10$ billion in 2013. It should be pointed that North America and Europe contribute to 70% of global revenue (2).

The adoption of wearable technology for health and medical applications has created a global market valued at 6.8$ billion the last 2015. Revenue in 2016 is expected to reach 7.3$ billion, and will reach 8.7$ billion in 2019, equivalent to

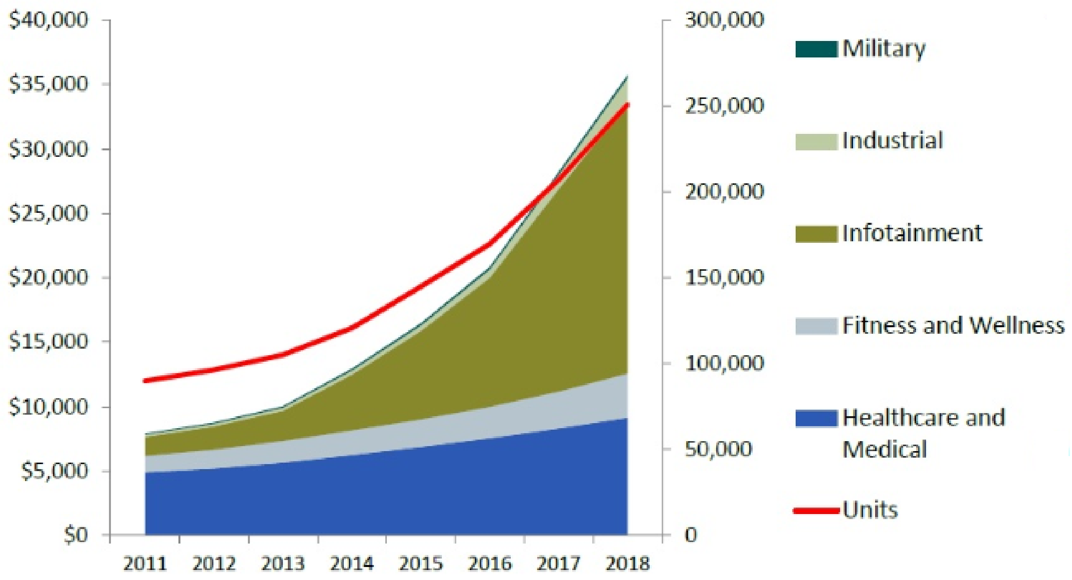


Figure 1.5: Global market trends or wearable technology in function of US millions and units millions. Figure obtained from (2)

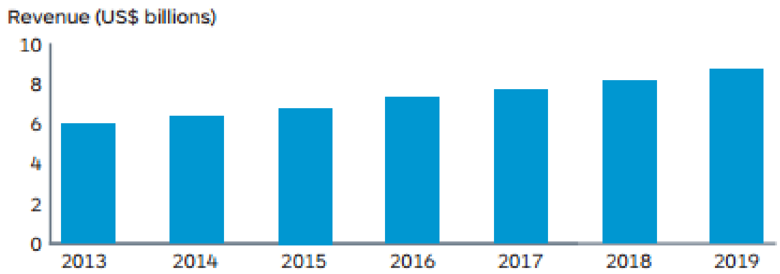


Figure 1.6: Global revenue forecast for wearable technology in health and medical applications. Figure reprinted from (3)

a compound annual growth rate of more than 6% starting from 2013. This is showed in Figure 1.6.

Smartphone adoption is also other important point to treat in this context. In 2016, over 1.2 billion smartphones are forecast to be shipped worlwide as the smartphone market experiences rapid growth. Therefore, as the number of smartphone users grows and the smartphone becomes the hub of information for its users, there will be an increasing number of devices that will connect to and exchange data with smartphones (2).

1.3 Objectives

According to the present favorable context mentioned in the preceding paragraphs and continuing with the development and the study of mDurance, this work will treat the next goals:

- The development of a new service of back-end to store all the data streamed by the sensors during the tests and an API Rest application to connect an Android device with the server.
- The migration of the old local database to a new storage system in the cloud.
- Add a new approach about the measurement of the endurance and muscle fatigue in relation to new methods based in time sliding window.
- Assessment of muscle activity in the lower back muscles through endurance tests on a football team applying known clustering and data mining techniques.

1.4 Structure

This document is divided into six chapters:

- **Chapter 1. Introduction**. It is the current chapter which is composed by four sections. The first section titled *Motivation and context* 1.1 describes the importance and need that urges in modern world to develop biomedical applications, that is the subject of this thesis. The second section, *Objectives* 1.3, contains an overall summary of the proposal objectives at the beginning of the thesis. Finally, the last section is *Thesis and Structure* 1.4 and presents the structure and composition of the thesis.

- **Chapter 2. State of the Art**. This chapter presents a review of four fields related to biomedical engineering: biomedical and electromyography applications Section 2.1, portable electromyography devices Section 2.2, data mining techniques Section 2.3 and electromyography studies based in data mining techniques Section 2.4.

- **Chapter 3. Methodology**. This chapter is divided into two parts. In the first part, it is reviewed the methodology followed in the realization of the previous final career project and it will be mentioned its most important features: Trunk endurance assessment Section 3.1 where it is described the different tests that they are used in the system and how it is assessed the muscle fatigue Section 3.2. Thereupon, they are explained the new improvements that the system incorporate in order to understand future chapters. These new improvements are divided into two levels. In a first system level, in which they are incorporated new services: a new back-end and API Rest service which are supported in a platform as service called Heroku 3.3. Both services are based in Python 3.3.1 and manage a new relational database based in PostgreSQL 3.3.2. In a second application level 3.4 should be pointed out that it has been developed a new cloud storage system related with the back-end service mentioned in the previous level; a new way to measure the muscle activity through a sliding windows mechanism based in temporal windows.

- **Chapter 4. Project Implementation**. In this chapter, it is described how the new main modules mDurance explained in the previous chapter have been implemented. Along the Section 4.1 is explained the development of the back-end and of the API Rest service: the specialist application 4.1.1, the patient application 4.1.2, the results data application 4.1.3 and the tests data application 4.1.4. In addition, in this last section will be presented the configuration of PostgreSQL and Heroku with Django. Finally, in the last section 4.2 is developed the implementation of an API Restful with Android.

- **Chapter 5. Classification of muscle activity in the lower back muscles**. This chapter is divided into two main sections. Initially, it is presented in Section 5.1 a description about the tests assessment in a professional football team. In Section 5.2, are exposed the results obtained in three subsections: a first statistical approach in the subsection 5.2.1; Then, it is presented a clustering analysis 5.2.2 to check the possible data clusters by a study based in their features; Last step is to study the different known classifiers 5.2.3 and to apply them to the previous clusters established.

- **Chapter 6. Conclusion and future work**. This chapter contains the conclusions reached in this thesis and proposal of future work 6.1.

2

State of the Art

In this chapter we describe the present and future investigation lines about electromyography and biomedical applications Section 2.1, portable electromyography devices Section 2.2, data mining techniques Section 2.3 and electromyography studies based in data mining techniques Section 2.4.

2.1 Electromiography applications

The applications of the biomedical engineering have a range so wide that it could be the topic of a thesis. However, this study is aimed to approach a review about applications where the electromyography is useful. Therefore this is just a description of the most relevant aspects. When it is talked about electromyography applications, it is referred to that field of study which mixes physiotherapy with computer science, electronics and other similar fields. The main goal of this subject is to develop the necessary technology for improving and making easier everything related to health, sport and wellness.

EMG has been a subject of laboratory research for decades. Only with recent technological developments in electronics and computers has surface EMG emerged from the laboratory as a subject of intense research in particularly kinesiology, rehabilitation and occupational and sports medicine. Most of the applications of surface EMG are based on its use as a measure of activation timing of muscle, a measure of muscle contraction profile, a measure of muscle contraction strength, or as a measure of muscle fatigue (36). Only a handful of research

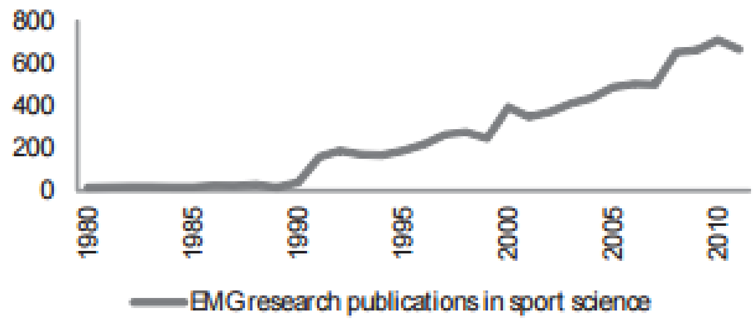


Figure 2.1: The growth in the number of EMG related publications in sport science since 1980s. Figure obtained from (4)

articles using EMG techniques were published in the early 1950s. Today, over 2500 research publications appear each year. The growth of the EMG literature and the availability of appropriate instrumentation and techniques might suggest that our understanding of the procedures used to record the EMG signal and the relevant analysis methods must be complete. Yet the interpretation of the signal remains controversial; and there are few sources available to help the novice electromyographer understand the physiological and biophysical basis of EMG, characteristics of the instrumentation, signal analysis techniques, and appropriate EMG applications (37).

The studies that use surface EMG in sciences of sports are mostly related with determination of the mechanism of contraction and relaxation of muscles while also dealing with evolution of injuries. The data obtained from these studies can be used in the following areas (4):

- The evaluation of the technical development
- The establishment of the suitable exercise programs
- Follow up of the development of the sportsmen
- The choice of skills.

Due to the specifity, it is complex to find EMG applications in the main application catalogs, i.e., Google Play and Apple Store because they requires

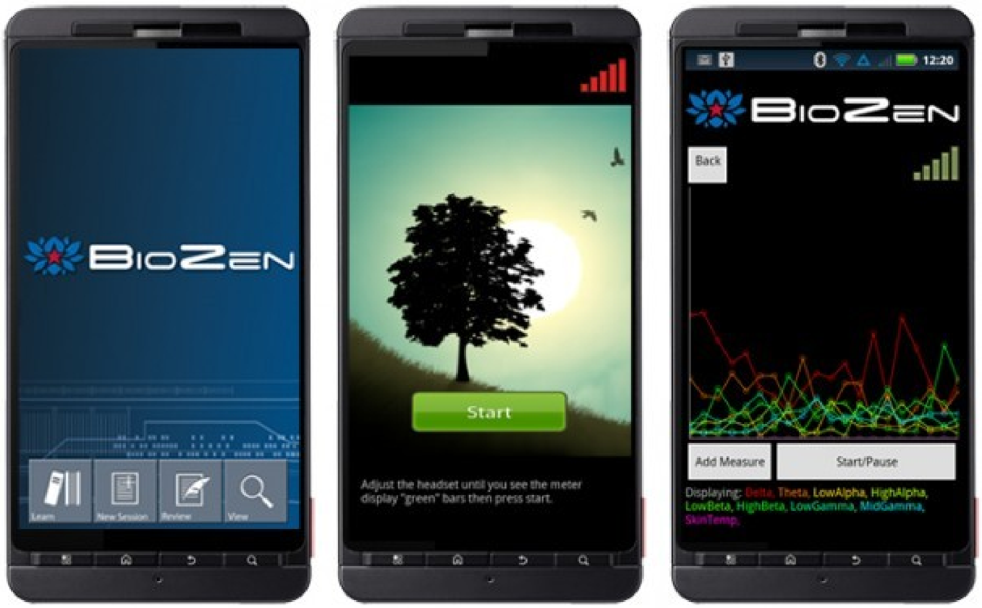


Figure 2.2: BioZen, an low-cost system for clinicians and patients to use biofeedback in and out of the clinic (5).

of EMG devices to can use it. Several apps can be found in relation to EMG applications in their own webs or in academics articles. The vast majority of apps are planned to provide an analysis about the muscle activity. Also, apps with informative or academic purposes and others focused on diagnosis are available. Some examples are:

- *BioZen* (5). It is treated of an Android smartphone app to help to the specialist to use the therapeutic benefits of biofeedback. BioZen is a mobile app from the Defense Departments National Center for telehealth and technology, uses Bluetooth-coupled sensors to show the user their physical level of relaxation. It is a portable, low-cost method for clinicians and patients to use biofeedback in and out of the clinic. BioZen shows real-time data from multiple body sensors including electroencephalogram (EEG), electromyography (EMG), galvanic skin response (GSR), electrocardiogram (ECG or EKG), respiratory rate, and skin temperature. Each sensor sends a separate signal to the phone so users can see how their body is responding to their behavior.

- *Real-time ECG and EMG Android System for biking* (38). This is an application for Android based mobile devices that enables a real-time calculation of heart rate and cadence for biking. The ECG algorithm is based on the PanTompkins algorithm for QRS-Detection and offers a heart beat detection rate of more than 94%. The EMG algorithm offers a treadle detection rate of more than 91%.

- *MIThril LiveNet* (39). This system is a flexible distributed mobile platform that can be deployed for a variety of proactive healthcare applications. The LiveNet system allows people to receive real-time feedback from their continuously monitored (by EKG/EMG/GSR sensors) and analyzed health state, as well as communicate health information with care givers and other members of an individuals social network for support and interaction.

- *Application for detect epileptic seizures using wearable sensors* (40). Epileptic seizures usually consist of stereotyped motor movements in association with characteristic changes in the electroencephalogram (EEG). Ongoing work by EEG sensor, the system uses wearable sensors to collect data from accelerometer and electromyography (EMG) probes over extended periods of time thus allow to assess the likelihood of false detection of seizure events when subjects perform activities of daily living. This evaluation to automatically analyze the recordings from wearable sensors with the aim of identifying patterns of motion and muscle activation that are associated with movements that accompany seizure events.

- *EMGNeu: Mobile health application for neuromuscular disorders diagnosis* (41). EMGNeu is a mobile health application that aims to help patients to discover their disease once it occurs based on EMG signal. The application is based on the wavelet transform approach for features extraction process and SVM model (it will be explained in next section) for classification process.

2.2 Portable electromyography devices

The design and implementation of wearable electromyography systems for health monitoring has gotten lots of attention throughout the world, specially in the sport, the physiotherapy and scientific community during the last years.

To address this demand, a variety of systems and commercial products have been produced in the course of recent years, they propelled by all the new technological advances like smart textiles, microelectronics, miniature bio-sensing devices and wireless communications. The continuous advance of wearable sensor-based systems will potentially transform the future of healthcare by enabling proactive personal health management and ubiquitous monitoring the electrical muscle activity to assess the patients health condition. These systems can comprise various types of small physiological sensors, transmission modules and processing capabilities, and can thus facilitate low-cost wearable unobtrusive solutions for continuous all-day and any-place health and activity status monitoring (42). These electromyography sensors are capable of measuring significant physiological parameters like muscular fatigue, posture analysis, strength, etc. Some examples of devices than can display these parameters are:

- *Shimmer EMG*. The Shimmer EMG measures and records the electrical activity associated with muscle contractions, assesses nerve conduction, muscle response in injured tissue, activation level, or can be used to analyze and measure the biomechanics of human or animal movement. The Shimmer EMG is non-invasive (Surface) EMG and therefore is a representation of the activity of the whole muscle. It is an efficient wireless solution for access to a host of muscle, gait, and posture data analysis. It is avalaible for $448 (43).

- *Athos*. It is a company based in California, USA, who are working on a product to gather and visualize biometric data using sensors in clothing, which they call smart apparel. The idea is to measure muscle performance, heart and breathing rates by a smart shirt and shorts (with 12 and 8 EMG sensors respectively) in real time and send this data to a smart phone for visualization. Depending on the wearable, the prizes can be different. For

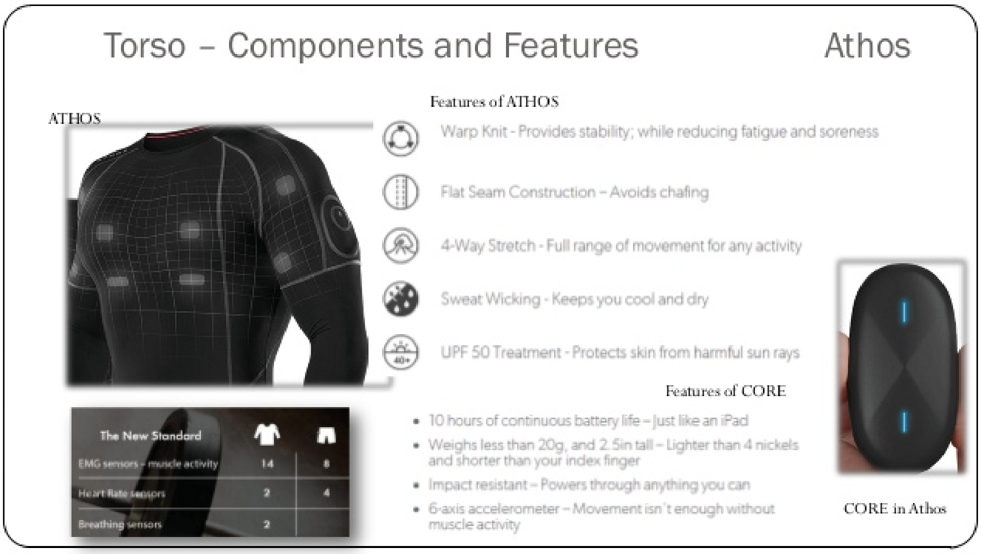


Figure 2.3: Components and features of Athos (6)

example, the full body pack costs $547, the upper body and lower body cost $398 and $348 respectively (6).

- *Myontec Mbody.* A system which can monitor your muscle load, balance and other variables in real time using a smart shorts and the mobile phone. Myontec Mbody translates the bio-signals into simple user interface and analyses. Recording exercise is made easy without the mobile device also. It is could to download the exercise afterwards into cloud service, analyze and compare its progress. The prize for this wearable is $820 (7).

- *DataLOG MWX8 and DataLINK.* They are overall, a system for data acquisition technology developed to meet the needs of researchers for portable data collection and monitoring in human performance, sports science, medical research, industrial ergonomics, gait laboratories, and educational settings. DataLINK is an on-line general purpose subject worn programmable Data Acquisition System allowing the user to collect both analog and digital data from a wide range of sensors including biometrics' goniometers, torsiometers, active EMG sensors, accelerometers, pinchmeters, hand dynamometers and Contact Switches (44).

- *Myo armband.* Thalmic Labs, based in Kitchener, Canada, is a company who has developed a gesture control armband called Myo that can sense electrical activity in the forearm in order to identify gestures. The activity can be transmitted to a computer, a smart phone, or any other Bluetooth compatible product. It can be obtained for $150 (45).

- *MyoWare Muscle Sensor.* It is an Arduino-powered, all-in-one electromyography sensor from Advancer Technologies. Its main advantage is that it is the cheapest sensor in the market. It can be bought for $38 (46).

- *Vidameter.* The 9 sensors of the VIDAMETER: IMU (9-axis inertial measurement unit) EMG Sensor (proprietary muscle activity, Temperature, Proximity, Altimeter, Oxygen, Ultrasonic, GPS, Pulse) are able to recognize an emergency from changes in users vital signs. If the device detects an injury or emergency, it automatically makes an emergency call to up to ten pre-defined contacts. Further, these emergency calls contain location data and information about the physical condition of the person in need, so that emergency workers lose little time and can take the necessary action immediately. Whats more, the VIDAMETER can detect and provide early warning for sleep problems and widespread diseases like Hypertension or Diabetes (47).

- *Vylyv.* This is another case where you can see how electromyography devices can be used for many different fields. VylyV claims the shorts will "peak the sexual performance by pelvic floor training and guard the manhood wellness against the sedentary lifestyle." The shorts, which are equipped with sensors and a wireless tracker, also come with a companion app. The shorts are being crowdfunded (48).

2.3 Data mining techniques

In this section, it is going to be deepen in the different techniques to treat the information obtained through the different devices.

Figure 2.4: Components of Myo armband (7)

Clustering is the unsupervised classification of patterns (observations, data items, or feature vectors) into groups (clusters). The clustering problem has been addressed in many contexts and by researchers in many disciplines. This reflects its broad appeal and usefulness as one of the steps in exploratory data analysis. However, clustering is a difficult problem combinatorially, and differences in assumptions and contexts in different communities has made the transfer of useful generic concepts and methodologies slow to occur.

In the first part of this section is wanted to present an overview of pattern clustering methods from a statistical pattern recognition perspective, with a goal of providing useful advice and references to fundamental concepts accessible to the broad community of clustering practitioners (49). It is important to understand the difference between clustering (unsupervised classification) and discriminant analysis (supervised classification). In supervised classification, it is provided with a collection of labeled (preclassified) patterns. The problem is to label a newly encountered, yet unlabeled, pattern. Typically, the given labeled (training)

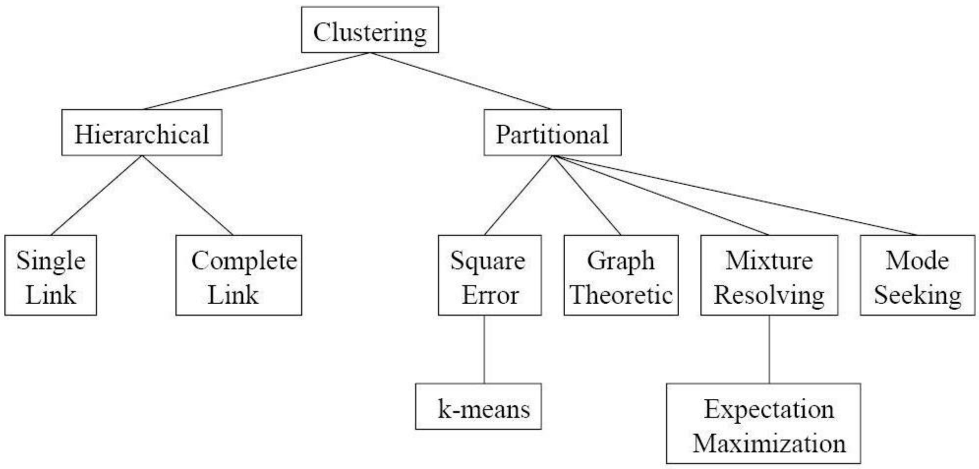


Figure 2.5: A taxonomy of clustering approaches. Figure obtained from (8).

patterns are used to learn the descriptions of classes which in turn are used to label a new pattern. In the case of clustering, the problem is to group a given collection of unlabeled patterns into meaningful clusters. In a sense, labels are associated with clusters also, but these category labels are data driven (49, 50).

Different approaches to clustering data can be described with the help of the hierarchy shown in Figure 2.5.

At the top level, there is a distinction between hierarchical and partitional approaches. The hierarchical methods produce a nested series of partitions, while the partitional methods produce only one.

The most hierarchical clustering algorithms are variants of the single-link, complete-link algorithms. These two algorithms differ in the way they characterize the similarity between a pair of clusters. In the single-link method, the distance between two clusters is the minimum of the distances between all pairs of patterns drawn from the two clusters (one pattern from the first cluster, the other from the second) (51). In the complete-link algorithm, the distance between two clusters is the maximum of all pairwise distances between patterns in the two clusters (52). In either case, two clusters are merged to form a larger cluster based on minimum distance criteria. The complete-link algorithm produces tightly bound or compact clusters. The single-link algorithm, by contrast, suffers from a chaining effect. It has a tendency to produce clusters that are straggly or elongated (53).

A partitional clustering algorithm obtains a single partition of the data instead of a clustering structure. A problem accompanying the use of a partitional algorithm is the choice of the number of desired output clusters (49). The partitional techniques usually produce clusters by optimizing a criterion function defined either locally (on a subset of the patterns) or globally (defined over all of the patterns). Combinatorial search of the set of possible labelings for an optimum value of a criterion is clearly computationally prohibitive. In practice, therefore, the algorithm is typically run multiple times with different starting states, and the best configuration obtained from all of the runs is used as the output clustering (54).

The most intuitive and frequently used criterion function in partitional clustering techniques is the squared error criterion, which tends to work well with isolated and compact clusters. The k-means is the simplest and most commonly used algorithm employing a squared error criterion. It starts with a random initial partition and keeps reassigning the patterns to clusters based on the similarity between the pattern and the cluster centers until a convergence criterion is met (e.g., there is no reassignment of any pattern from one cluster to another, or the squared error ceases to decrease significantly after some number of iterations). The k-means algorithm is popular because it is easy to implement (55). The mixture resolving approach to cluster analysis has been addressed in a number of ways. The underlying assumption is that the patterns to be clustered are drawn from one of several distributions, and the goal is to identify the parameters of each and (perhaps) their number. The most of the work in this area has assumed that the individual components of the mixture density are Gaussian, and in this case the parameters of the individual Gaussians are to be estimated by the procedure. Traditional approaches to this problem involve obtaining (iteratively) a maximum likelihood estimate of the parameter vectors of the component densities (50).

More recently, the Expectation Maximization (EM) algorithm (a general purpose maximum likelihood algorithm (56) for missing-data problems) has been applied to the problem of parameter estimation. In the EM framework, the parameters of the component densities are unknown, as are the mixing parameters, and these are estimated from the patterns. The EM procedure begins with an initial estimate of the parameter vector and iteratively rescores the patterns against the mixture density produced by the parameter vector. The rescored patterns are then used to update the parameter estimates.

The clustering algorithms are attractive for the task of class identification in spatial databases. However, the application to large spatial databases rises the following requirements for clustering algorithms: minimal requirements of domain knowledge to determine the input parameters, discovery of clusters with arbitrary shape and good efficiency on large databases. The known clustering algorithms offer no solution to the combination of these requirements. In this context, (57) proposed DBSCAN: an algorithm based in the analysis of density , which is designed to discover clusters of arbitrary shape. DBSCAN requires only one input parameter and supports the user in determining an appropriate value for i. The density is estimated for one concrete point and it is counted the number of points around of a closed context in the same and of a fixed radius. The density of each point depend on the radius that is had.

While the density based methods, particularly the non-parametric density based approaches, are attractive because of their inherent ability to deal with arbitrary shaped clusters, they have limitations in handling high-dimensional data. When the data is high-dimensional, the feature space is usually sparse, making it difficult to distinguish high-density regions from low-density regions. Subspace clustering algorithms overcome this limitation by finding clusters embedded in low-dimensional subspaces of the given high-dimensional data. CLIQUE (58) is a scalable clustering algorithm designed to find subspaces in the data with high-density clusters. Because it estimates the density only in a low dimensional subspace, CLIQUE does not suffer from the problem of high dimensionality. The traditional clustering approaches generate partitions. In a partition, each pattern belongs to one and only one cluster. Hence, the clusters in a hard clustering are disjoint. The fuzzy clustering extends this notion to associate each pattern with every cluster using a membership function (59). The output of such algorithms is a clustering, but not a partition. It is given a high-level partitional fuzzy clustering algorithm below. In this context, it exits partitional methods extensions, for example, the fuzzy c-means, proposed by (60) and later improved by (61), is an extension of K-means where each data point can be a member of multiple clusters with a membership value (soft assignment).

In the second part of this section they are going to be presented an overview of supervised classification techniques. In supervised classification, it is provided with a collection of labeled (preclassified) patterns. The problem is to label a newly encountered, yet unlabeled, pattern. Typically, the given labeled (training) patterns are used to learn the descriptions of classes which in turn are used to label a new pattern. Supervised classification is one of the task most frequently carried out by Intelligent System. Thus, a large number of techniques have been developed based on Artificial Intelligence and Statistics. The rest of this section, it will be focused on the most important supervised machine learning techniques:

The first important techniques to mention are concentrated on two groups of logic learning methods. On the one hand are *decision trees*. The decision trees are trees that classify instances by sorting them based on feature values. Each node in a decision tree represents a feature in an instance to be classified, and each branch represents a value that the node can assume. The instances are classified starting at the root node and sorted based on their feature values (62). The most well-know algorithm in the literature for building decision trees is the C4.5 (63). It has a very good combination of error rate and speed. To sum up, one of the most useful characteristics of decision trees is their comprehensibility. People can easily understand why a decision tree classifies an instance as belonging to a specific class. On the other hand are the *learning set of rules*. The decision trees can be translated into a set of rules by creating a separate rule for each path from the root to a leaf in the tree (63). However, rules can also be directly induced from training data using a variety of rule-based algorithms. The goal is to construct the smallest rule set that is consistent with the training data. A large number of learned rules is usually a sign that the learning algorithm is attempting to remember the training set, instead of discovering the assumptions that govern it. A separate-and-conquer algorithm search for a rule that explains a part of its training instances, separates these instances and recursively conquers the remaining instances by learning more rules, until no instances remain (64). It is therefore important for a rule induction system to generate decision rules that have high predictability or reliability. These properties are commonly measured by a function called rule quality (65). RIPPER is a well-known rule-based algorithm (66). It forms rules through a process of repeated growing and pruning. During the growing phase the rules are made more restrictive in order to fit the training data as closely as possible. During the pruning phase, the rules are made less restrictive in order to avoid over-fitting, which can cause poor performance on unseen instances. RIPPER handles multiple classes by ordering them from least to most prevalent and then treating each in order as a distinct two-class problem. Genetic algorithms (GAs) have also been used for learning sets of rules. In this study (67), it is used a genetic algorithm to learn binary concepts represented by a disjunctive set of propositional rules and it was found to be comparable in generalization accuracy to other learning algorithms. To sum up, the most useful characteristic of rule-based classifiers is their comprehensibility. In addition, even though some rule-based classifiers can deal with numerical features, some experts propose these features should be discretized before induction, so as to reduce training time and increase classification accuracy (68).

Other well-known algorithms are based on the notion of *perceptron*. Perceptron can be briefly described as a continuous inputs and binary outputs set. Each input has an associated weight. Positive and negative weights correspond to excitatory and inhibitory inputs. If the sum is above threshold, output is 1; else it is 0. The most common way the perceptron algorithm is used for learning from a batch of training instances is to run the algorithm repeatedly through the training set until it finds a prediction vector which is correct on all of the training set. This prediction rule is then used for predicting the labels on the test set (62). Perceptrons can only classify linearly separable sets of instances. If a straight line or plane can be drawn to seperate the input instances into their correct categories, input instances are linearly separable and the perceptron will find the solution. If the instances are not linearly separable learning will never reach a point where all instances are classified properly. *Artificial Neural Networks* have been created to try to solve this problem. In (69) is provided an overview of existing work in Artificial Neural Networks (ANNs). A multi-layer neural network consists of large number of units (neurons) joined together in a pattern of connections (Figure 2.6). The units in a net are usually segregated into three classes: input units, which receive information to be processed, output

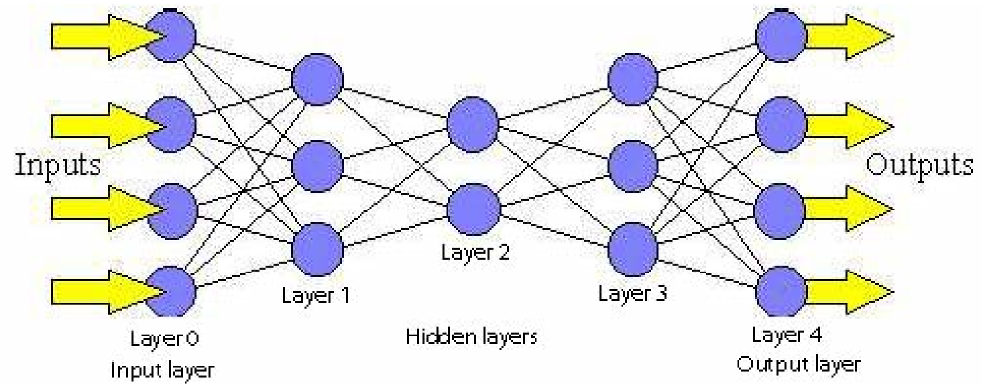


Figure 2.6: Feed-fordward ANN.

units, where the results of the processing are found and units in between known as hidden units.

ANN depends upon three fundamental aspects, input and activation functions of the unit, network architecture and the weight of each input connection. Given that the first two aspects are fixed, the behavior of the ANN is defined by the current values of the weights. The weights of the net to be trained are initially set to random values, and then instances of the training set are repeatedly exposed to the net. The values for the input of an instance are placed on the input units and the output of the net is compared with the desired output for this instance. Then, all the weights in the net are adjusted slightly in the direction that would bring the output values of the net closer to the values for the desired output.

The *Radial Basis Function (RBF)* networks have been also widely applied in many science and engineering fields (70). An RBF network is a three layer feedback network, in which each hidden unit implements a radial activation function and each output unit implements a weighted sum of hidden units outputs. Its training procedure is usually divided into two stages. First, the centers and widths of the hidden layer are determined by clustering algorithms. Second, the weights connecting the hidden layer with the output layer are determined by Singular Value Decomposition (SVD) or Least Mean Squared (LMS) algorithms. The problem of selecting the appropriate number of basis functions remains a critical issue for RBF networks. The number of basis functions controls the complexity and the generalization ability of RBF networks. RBF networks with too few basis functions cannot fit the training data adequately due to limited flexibility. On the other hand, those with too many basis functions yield poor generalization abilities since they are too flexible and erroneously fit the noise in the training data. To sum up, ANNs have been applied to many real world problems but still, their most striking disadvantage is their lack of ability to reason about their output in a way that can be effectively communicated. For this reason many researchers have tried to address the issue of improving the comprehensibility of neural networks, where the most attractive solution is to extract symbolic rules from trained neural networks. Conversely to ANNs, statistical approaches are characterized by having an explicit underlying probability model, which provides a probability that an instance belongs in each class, rather than simply a classification. Under this category of classification algorithms, one can find *Bayesian networks* and *instance-based methods*.

A *Bayesian Network (BN)* is a graphical model for probability relationships among a set of variables (features). Typically, the task of learning a Bayesian network can be divided into two sub-tasks: initially, the learning of the structure of the network, and then the determination of its parameters. Probabilistic parameters are encoded into a set of tables, one for each variable, in the form of local conditional distributions of a variable given its parents. Given the independence encoded into the network, the joint distribution can be reconstructed by simply multiplying these tables. Within the general framework of inducing Bayesian networks, there are two scenarios: known structure and unknown structure (71).

Instance-based learning algorithms are lazy-learning algorithms (72), as they delay the induction or generalization process until classification is performed. Lazy-learning algorithms require less computation time during the training phase than other learning algorithms but more computation time during the classification process.

One of the most well-know instance-based learning algorithms is the nearest neighbor algorithm. *k-Nearest Neighbor (kNN)* is based on the principle that the instances within a dataset will generally exist in close proximity to other instances that have similar properties. If the instances are tagged with a classification label, then the value of the label of an unclassified instance can be determined by observing the class of its nearest neighbors. The kNN locates the k nearest instances to the query instance and determines its class by identifying the single most frequent class label.

Support Vector Machines (SVMs) are the newest supervised machine learning technique. An excellent survey and tutorial of SVMs can be found in (73). SVMs revolve around the notion of a margin either side of a hyperplane that separates two data classes. Maximizing the margin and thereby creating the largest possible distance between the separating hyperplane and the instances on either side of it has been proven to reduce an upper bound on the expected generalization error. In the case of linearly separable data, once the optimum separating hyperplane is found, data points that lie on its margin are known as support vector points and the solution is represented as a linear combination of only these points. Other data points are ignored. Therefore, the model complexity of an SVM is unaffected by the number of features encountered in the training data (the number of support vectors selected by the SVM learning algorithm is usually small). For this reason, SVMs are well suited to deal with learning tasks where the number of features is large with respect to the number of training instances. Even though the maximum margin allows the SVM to select among multiple candidate hyperplanes, for many datasets, the SVM may not be able to find any separating hyperplane at all because the data contains misclassified instances. The problem can be addressed by using a soft margin that accepts some classifications of the training instances (74). With an appropriately chosen feature space of sufficient dimensionality, any consistent training set can be made separable. A linear separation in feature space corresponds to a non-linear separation in the original input space. Mapping the data to some other by a kernel function. Thus, kernels are a special class of function that allow inner products to be calculated directly in feature space, without performing the mapping described above (75). Once a hyperplane has been created, the kernel function is used to map new points into the feature space for classification. The selection of an appropriate kernel function is important, since the kernel function defines the feature space in which the training set instances will be classified. In (76) is described several classes of kernels.

Numerous methods have been suggested for the creation of ensemble of classifiers (77). Thus, it is treated of an active area of research in supervised learning is the study of methods for the construction of good ensembles of classifiers. The mechanisms that are used to build ensemble of classifiers include: using different subsets of training data with a single learning method, using different training parameters with a single training method (e.g., using different initial weights for each neural network in an ensemble) and using different learning methods. In the first class is founded Bagging. This is a method for building ensembles that uses different subsets of training data with a single learning method (78). Given a training set of size t, bagging draws t random instances from the dataset with replacement (i.e. using a uniform distribution). These t instances are learned, and this process is repeated several times. Since the draw is with replacement, usually the instances drawn will contain some duplicates and some omissions, as compared to the original training set. Each cycle through the process results in one classifier. After the construction of several classifiers, taking a vote of the predictions of each classifier produces the final prediction.

Another method that uses different subsets of training data with a single learning method is the *boosting* approach (79). *Boosting* is similar in overall structure to bagging, except that it keeps track of the performance of the learning algorithm and concentrates on instances that have not been correctly learned. Instead of choosing the t training instances randomly using a uniform distribution, it chooses the training instances in such a manner as to favor the instances that have not been accurately learned. After several cycles, the prediction is performed by taking a weighted vote of the predictions of each classifier, with the weights being proportional to each classifiers accuracy on its training set. *AdaBoost* is a practical version of the boosting approach (79). Adaboost requires less instability than bagging, because Adaboost can make much larger changes in the training set. A number of studies that compare AdaBoost and bagging suggest that AdaBoost and bagging have quite different operational profiles (80, 81). In general, it appears that bagging is more consistent, increasing the error of the base learner less frequently than does AdaBoost. However, AdaBoost appears to have greater average effect, leading to substantially larger error reductions than bagging on average. Another novel algorithm is *XGBoost*, a scalable machine learning system for tree boosting (82). The system is available as an open source package. The impact of the system has been widely recognized in a number of machine learning and data mining challenges. To sum up, if it is only interested in the best possible classification accuracy, it might be difficult or impossible to find a single classifier that performs as well as a good ensemble of classifiers.

2.4 Electromyography studies based in data mining techniques

Recent advances in wearable sensor technology mentioned in Section 1 and the proliferation of new devices mentioned in Section 2.2 indicate that numerous systems are currently available or under development that could be utilized to monitor personal health and the electrical muscle activity to assess the patients health condition. Also, the interest in wearable systems is growing and is expected to lead in the near future to systems that can be worn continuously like a normal garment. Techniques that take full advantage of the extraordinary amount of data that such sensors/systems can gather are lacking. Several analysis provides a very large data volume coming from electromyographic registers and physical examinations. The analysis and treatment of these data is difficult and time consuming.This section explores exhaustively different analysis methods from data mining on these analysis data for different fields.

- *Data mining techniques to detect motor fluctuations in Parkinson's disease* (83). The purpose of this work is to present preliminary evidence that data mining and artificial intelligence systems may allow one to recognize the presence and severity of motor fluctuations in patients with Parkinsons disease (PD).

- *Data mining techniques on EMG registers of hemiplegic patients* (84). This study aims to provide a classification system based in gait patterns obtained from EMG records in children with spastic hemiplegia. The methods studied from data mining specifically for the classification task include SVM, neural networks, decision trees, regression logistic models and others. Different techniques of feature extraction and selection have been also employed and combined with classifications methods.

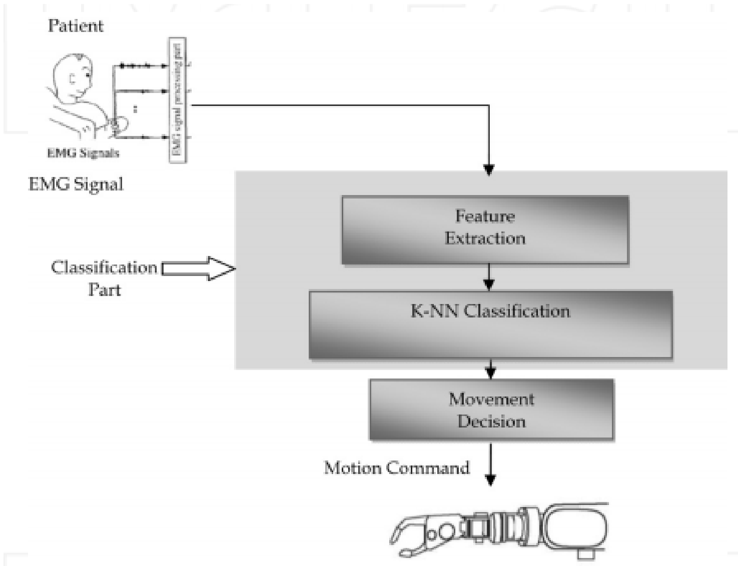


Figure 2.7: Structure of the arm movement recognition system based on K-NN classifier. Figure reprinted from (9).

- *Classification of EMG signals using PSO optimized SVM for diagnosis of neuromuscular disorders* (85). In this study, a novel PSO-SVM model has been proposed that hybridized the particle swarm optimization (PSO) and SVM to improve the EMG signal classification accuracy. This optimization mechanism involves kernel parameter setting in the SVM training procedure, which significantly influences the classification accuracy. The experiments were conducted on the basis of EMG signal to classify into normal, neurogenic or myopathic.

- *Electromyography of selected lower-limb muscles fatigued by exercise at the intensity of soccer match-play.* This study aimed to investigate the activity of major muscles of the lower extremity during a soccer simulation fatiguing protocol. Each muscle was analyzed by means a two-factor (fatigue condition x speed) repeated measures analysis of variance (ANOVA) with the least significant difference test (LSD) used where EMG was recorded at three time-points (pre-exercise, half-way and post-exercise) and four speeds (running at 6, 12, 15 and 21 km/h).

- *Analysis of human arm motions recognition algorithms for system to visualize virtual arm* (86). This paper presents preliminary studies on the problem of classification of different kinds of human arm motions based on EMG signals. It is considered some problems related to plan routine and monitor tennis training. It is also considered following algorithms to solve the problem of classification mentioned: C 4.5 algorithm and Repeated Incremental Pruning to Produce Error Reduction (RIPPER). Additionally, in the studies it also test multilayer Perceptron (MLP) and one of the most common ensemble classifier, bagging predictor with classification and regression trees as base classifiers.

3

Methodology

This chapter is divided into two parts. In the first part, it is reviewed the methodology followed in the realization of the previous work (10) and it will be mentioned its most important features: Trunk endurance assessment Section 3.1 where it is described the different tests that they are used in the system and how it is assessed the muscle fatigue Section 3.2. Thereupon, the new improvements that the system incorporate are explained in order to understand future chapters. These new improvements are divided into two levels. In a first system level, in which they are incorporated new services: a new back-end service 3.3 based Python 3.3.1 and that it manages a new relational database based PostgreSQL 3.3.2. They are supported in a platform to deploy and run the software applications called Heroku. A new API RestFul also developed with Python for support Android applications and can communicate Android devices with the back-end system. In a second application level 3.4 should be pointed out that it has been developed a new cloud storage system related with the back-end service mentioned in the previous level. A new way to measure the muscle activity through a sliding windows mechanism based temporal windows.

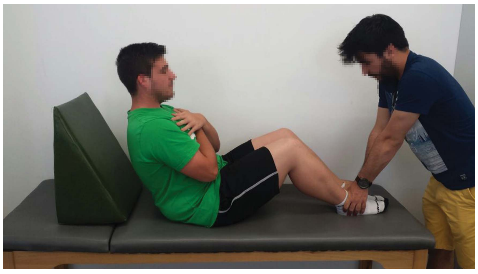
3.1 Trunk endurance assessment

This section is important for to remember the useful of the system designed in the previous work. It is important to know, what the endurance tests are and why they are used. It should be pointed out that for to assess the endurance and the muscle activity in the low back in the study treated in the future sections, it is important understand the working of these tests.

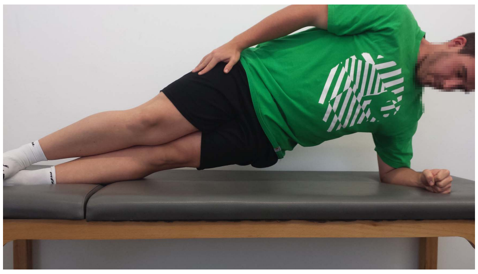
Different tests are available to assess the trunk endurance in people with or without LBP. These kind of tests are performed by a specialist, and they normally consist in the measurement of the time a person can hold a specific posture involving the trunk muscles. During the execution of the test, the health professional has to control the patient position and decide when the test ends, according to some established termination criteria. The results obtained for a given patient help experts determine their status and muscular capacity, as well as their ability to hold a posture normally related to daily living activities. To assess the low back stabilization several functional trunk endurance tests can be found in the literature (87, 88). The most widely used ones are the *static trunk extensor endurance test* (STEET), also known as Sorensen test (89), the *trunk curl static endurance test* (TCSET), also known as trunk flexor endurance test (90), and the *side bridge endurance test* (SBET) (91) (see Figure 3.1). In the STEET, the subject has to maintain a horizontal unsupported posture with the upper body extending beyond the edge of the bench. In the TCSET, a curled position must be hold with only the scapulae clearing the table. Finally, the SBET requires the individual to lie on their side while lifting the torso and thigh off the bench, such that the body weight is on the elbow and feet. Special remarks are that two chances are given to the individual to execute the STEET, while evaluation of both left and right sides are considered as part of the SBET. The position is held up to a maximum of 240s. A detailed description of each test, including posture, procedure and finalization criteria, is shown the previous work (10, 11)

Based on the regular experience of specialists, practical limitations can be observed during the course of the realization and evaluation of endurance tests. First of all, it is generally accepted that the tester has an important responsibility while determining the different phases of the test. The estimation of the beginning and end of the tests is completely subject to the expert visual interpretation. In fact, specialists often report on the difficulties faced during the observation of the trunk angle variation and termination criteria (92). Moreover, according to the nature of the tests (93, 94), the specialist normally needs to control several

(a)



(b)



(c)

Figure 3.1: Functional trunk endurance tests: (**a**) STEET; (**b**) TCSET; and (**c**) SBET.

aspects simultaneously, such as, time, position, and possible abnormalities during the test, which in traditional procedures could be despised. Finally, the results are mainly elaborated on the time recorded during the performance of the test, and that is the unique information to compare with in future tests. These drawbacks make complex the comparison of values measured by different testers and among sessions.

3.2 Muscle fatigue: analysis and description

In this section, it is explained importantly, what is the EMG and the meaning of muscle fatigue and we describe the different techniques to measure the EMG signal and how we can use that signal to extract the most important features to characterize the muscle fatigue.

According to (95), electromyography (EMG) is based on the study of muscle activity through observation and analysis of the electrical signals monitored during voluntary or involuntary muscle contractions. EMG is useful to study of muscular function during activities of sports, biofeedback training, daily living or detect pathological states of the musculoskeletal systems. In order to assess all those functions, EMG offers useful information about timing of muscular activity and its relative intensity.

For (96), muscle fatigue is manifested as a reduction in the ability to keep a certain level of strength in a sustained contraction or as the inability to achieve a level of initial strength in intermittent contractions and it is accompanied by changes in muscle electrical activity. Muscle fatigue has central and peripheral components. The first are manifested as an inability to realize an induced activity for the development of this. The second stand out as a muscle inability to produce a certain level of strength. In applications of electrical stimulation it is important to assess the development of muscle fatigue to prevent deterioration of the mechanical behavior of the muscle being stimulated. Muscle fatigue can be evaluated by EMG signal. The effect of fatigue on muscle mechanics is well represented by various metrics, such as the mean frequency (MNF), median frequency (MDF), the root mean square (RMS), the rectified root mean square (RRMS), the average rectified value (ARV) and the maximum voluntary muscle contraction (MVC) (95, 97). This information is of much interest to compare the evolution of the muscle strength among sessions, as well as to measure the effectiveness of potential treatments. In the previous study (10, 11), it is monitored the EMG signal, and it is gave a time window or epoch of N samples then, the RMS, ARV and MVC values can be calculated as follows:

$$RMS = \sqrt{\frac{\sum\limits_{k=1}^{N} EMG^2\left(k\right)}{N}} \tag{3.1}$$

$$ARV = \frac{\sum\limits_{k=1}^{N} \left|EMG\left(k\right)\right|}{N} \tag{3.2}$$

$$MVC = max\left(EMG\left(k\right)\right) \tag{3.3}$$

However, for the analysis of the EMG signal in this project will be used a feature extraction method proposed by (98) for myoelectric control that uses multiple time windows (MTW). In this time-domain method, the change in signal energy is calculated with respect to time by using various multiple window functions. Such windows can accommodate the variations of the signal parameters due to the non-stationarity. Moreover, the energy calculations with square integral method uses the full signal and provides only a single feature, whereas with the MTW technique, multiple features can be obtained. Multiple features provide a more complete set of information than any single parameter (99).

Then, both the RMS value and the ARV value are going to be extracted with sliding windows. For both features are used a window with a size equal to the 10% of the total duration that a person is realizing a determinate test. Thus, the total window size is between 0.01 and 2.4 seconds. It is noteworthy that the sliding window has an overlap time of 50% to get information both of the present time window as the past time window, creating a robust method to study the muscle activity during the tests. In the time domain the RMS of the EMG signal is considered the most reliable parameter. An increase of the EMG RMS with advancing fatigue has been reported in many studies. RMS is not affected by the cancellation due to motor unit action potential train superposition, which may affect other processing techniques involving rectification (97). This study (95) showed that muscle fatigue was accompanied with an increase in the RMS and ARV and a decrease in the MNF.

3.3 Cloud computing back-end solution for the mobile application

For many software developers working on mobile applications, the features and resources available on mobile devices are not sufficient to provide the level of Quality of Experience (QoE) users have come to expect. Although, the current top of the market mobile devices come with high processing power (multiple core processors), large storage and RAM, mobile developers must take advantage of the high-speed data connections available and look for alternatives on how to provide a richer experience for their applications. Cloud Computing solutions provide essential tools for building modern mobile applications. In order to leverage the advantages of the Cloud for developing and scaling applications, mobile developers must perform a technical analysis of the options currently available on the market.

Cloud Computing provides a combination of service models such as: Software-as-a-Service (SaaS), Infrastructureas-a-Service (IaaS) and Platform-as-a-Service (PaaS). Paas provides hardware and software tools required for application development. As both hardware and software are hosted in the providers infrastructure, users are free to focus soley on the application development. PaaS in its nature was created to offer businesses and independent developers the flexibility to have a robust, reliable and scalable platform to deploy and run software applications without the usual necessary system administrator skills or a high-risk upfront monetary investment (100). However, there are multiple companies offering PaaS solutions on the market (e.g., Heroku (101), RedHat OpenShift (102), AWS Elastic Beanstalk (103)), ranging from starting plans free of cost to expensive enterprise plans. Therefore, it can be daunting for software developers to understand what is in fact offered for free, what are the options when making investments and which service offers the best performance. Although there are several reasons why Cloud Computing has become a valuable resource for mobile applications, one of the key reasons is the broad network access. That is, the combination of the Internet, advanced wireless technologies, and electronics, mobile devices are capable of being completely connected and constantly available (104).

Finally, tying this all together is the advent of simple, easily consumed, RESTful web-services. REST is an acronym for REpresentative State Transfer. It was first introduced in (105). Web-Services based on REST are easier to consume on mobile platforms because the client and server agree on a simple invocation and response protocol. This eliminates the requirement for excessive meta-data based parsing for invocations. Traditional web-services are memory and processor intensive, which does not naturally lend itself to the limited memory and processing of smart mobile device environment. To that end, RESTful Web Services lend themselves very nicely to smart mobile device environments. They are easy to invoke, produce a discretely formatted response, and can usually be easily parsed using event-driven XML or JSON parsing which is less memory intensive than tree based parsing (104). The key aspects of REST that make it desirable for mobile applications are these:

- REST is URL based, therefore easy to invoke.
- REST responses are usually HTTP based (106), therefore discrete and it also minimizes the impact of network volatility.
- REST delivery can be made very succinct, lends itself to constrained memory environments. It is not superfluous protocol elements.

Generating a REST based request is not tied to any particular technology, but for REST based web-services one commonly uses HTTP and the corresponding HTTP verbs HEAD, POST, PUT, GET, DELETE. REST based web-services over HTTP are what we will focus on since most service providers presently use this model. Ultimately, there is no established standard for invocation requirements other than what is agreed upon by the service provider and the client. As it is mentioned above, HTTP is the preferred method, and for mobile computing, it is ideal. The stateless, discrete, per-request nature of both REST and HTTP are ideal for volatile network environments. A discrete request lends itself well to event-driven JSON processing which, as we will discuss in the next section, is the preferred JSON processing approach on mobile devices.

3.3.1 Developing a REST web-service with Python and Django Rest Framework

Django is a free and open-source web framework, written in Python, which follows the model-view-template (MVT) architectural pattern (107). It is maintained by the Django Software Foundation (DSF). Django's primary goal is to ease the creation of complex, database-driven websites. Django emphasizes reusability and pluggability of components, rapid development, and the principle of don't repeat yourself. Python is used throughout, even for settings files and data models. Django also provides an optional administrative create, read, update and delete interface that is generated dynamically through introspection and configured via admin models. The main Django distribution also bundles a number of applications in its "contrib" package, including:

- An extensible authentication system.
- The dynamic administrative interface.
- A sites framework that allows one Django installation to run multiple websites, each with their own content and applications.
- Built-in mitigation for cross-site request forgery, cross-site scripting, SQL injection, password cracking and other typical web attacks, most of them turned on by default.
- A framework for creating Rest applications called Django Rest Framework. This is important because the API Rest is built on this framework. It will be talked about this in the next subsection.

Django can be run in conjunction with Apache, NGINX (108) using WSGI (109), Gunicorn (110) using flup (a Python module). It is also possible to use other WSGI-compliant web servers. Django officially supports four database backends: PostgreSQL, MySQL, SQLite, and Oracle. Microsoft SQL Server can be used with django-mssql on Microsoft operating systems,while similarly external backends exist for IBM DB2, SQLAnywhere, etc. There is a fork named django-nonrel, which supports NoSQL databases, such as MongoDB and Google App Engine's Datastore. On the other hand, for can to develop an API Rest service is used Django REST framework (111). It is a powerful and flexible toolkit for building Web APIs. These are several reasons to use REST framework:

- The Web browsable API is a huge usability win for your developers.
- Authentication policies including packages for OAuth1a and OAuth2.
- Serialization that supports both ORM and non-ORM data sources.
- Customizable all the way down based views if it does not need the more powerful features.

- Extensive documentation, and great community support.
- Used and trusted by internationally recognised companies including Mozilla, Red Hat, Heroku, and Eventbrite.

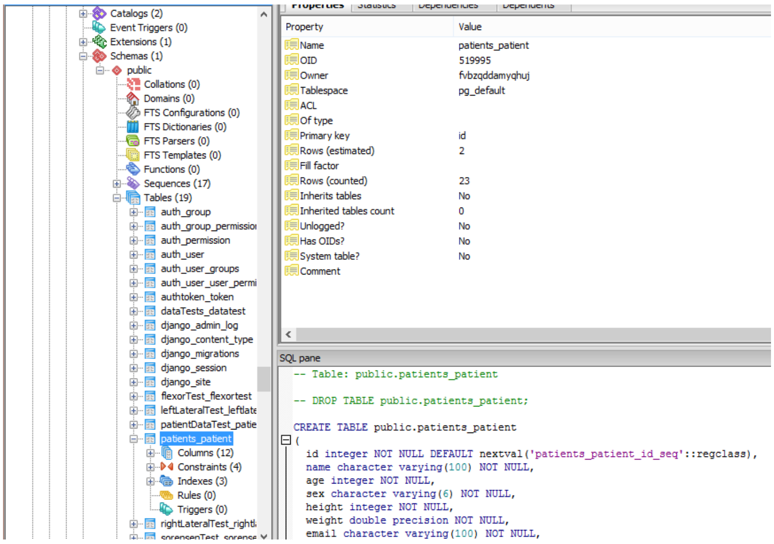
3.3.2 PostreSQL

PostgreSQL (112), unlike ORACLE that requires a license fee, is one of the open source relational database management system (RDBMS). Open source RDBMS is free of charge, which includes PostgreSQL, MariaDB (113), etc. However, PostgreSQL has been used, for its ease of installation. PostgreSQL implements the server/client model and supports the standard language of database, ANSI:SQL2011, and most of its functions including transactions. Due to separation of client and server in PostgreSQL, clients library became lighter, and any changes in the database engine will not affect the client (114).

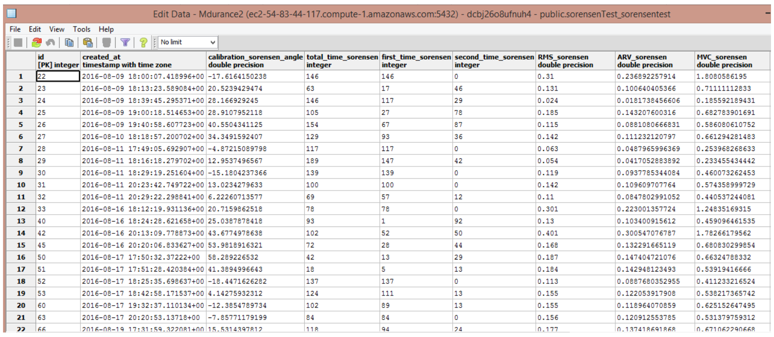
An interesting feature of PostgreSQL is the called multiversion concurrency control (MVCC). It is designed for environments with high volumes of traffic/-transactions. This method adds an image of the state of the database to each transaction. This allows to make eventually consistent transactions, offering advantages in the performance. For example, it does not requires to use read locks to make a transaction which gives a great scalability.

PgAdmin is the main database administrator Postgres and it is very easy to use. In the Figure 3.2 is showed the main views of PgAdmin. Virtually, this tool does it all with a few simple clicks, being optimal for people who do not have much experience, either because they are starting in the world of base data or because they simply need a database for your project. In turn it can be found options within PgAdmin where the most experienced people can handle their specific codes openly and feel comfortable with it.

Another feature about PostgreSQL is the Hot-Standby. This allows that the customers search (read-only) on servers while these are in a backup process or standby. In this way, it can be done maintenance or recovery without completely block the system.



(a)



(b)

Figure 3.2: PostgreSQL administrador (Pgadmin): (**a**) global view of mDurance classes; (**b**) detail view of the database of the Sorensen test class.

3.4 Application description

In this section, it is reviewed the working of the application developed in (10) in order to remember its main features and to introduce the new improvements that they have been introduced in the system.

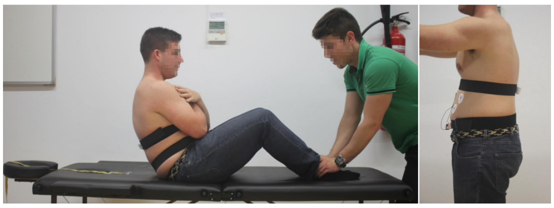
One of the main aims of the mDurance system is to help experts assess, in a precise manner, the time invested by the patients during the execution of the trunk endurance test, as well as the amount of muscle fatigue experienced in that process. To attain the first objective, an IMU sensor is considered to determine when the test termination criterion is met, based on the principle presented in Section 3.1.

For the second goal,an EMG sensor is used to continuously detect the electrical potential generated by the muscle cells in the course of the test, as explained in Section 3.2. Shimmer wearable sensors, concretely, version 2 for the EMG and version 3 for the IMU are employed, given the high reliability yielded by these commercial devices. The sampling rates that are used for both sensors are 15Hz for the IMU and 10Hz for the EMG, since they prove to be enough for an accurate estimation of the trunk angle and EMG metrics. Figure 3.3 shows the sensor deployment for each of the three trunk endurance tests supported by mDurance and described in Section 3.1. The sensors are located in convenient positions to ensure stability and comfortability, as well as an accurate measurement of both trunk angles and EMG values for each test. In the STEET and TCSET, the trunk angle is measured with respect to the coronal plane, while for the SBET the reference corresponds to the sagital plane. Accordingly, the IMU sensor is attached to the lumbar zone (D12-L1 vertebra) for the STEET and TCSET procedures, and to the dorsal for the SBET. Taking into account the placement of the IMU sensor for each case, and its local frame of reference orientation, the roll angle (ϕ) is used to represent the trunk angle in all tests.

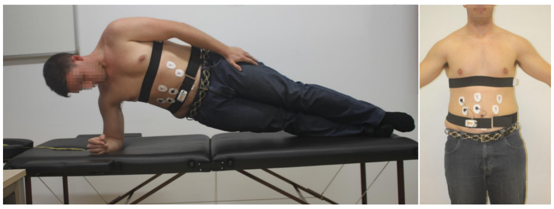
The EMG sensor is placed on the lumbar (erector spinae), abdominal (rectus abdominis) and external oblique parts for the STEET, TCSET, an SBET, respectively. The electrodes are distributed to cover a sufficient muscle area. Both IMU and EMG sensors are safely and firmly fastened to each corresponding body part through ergonomic straps that ensure no misplacement of the sensors. Moreover, there is little room for errors during the placement of the devices since the sensors must be positioned by the experts.

In the following the mDurance application is reviewed and it is proceeded to introduce the new improvements within the application level. For this first time use, the expert is requested to sign up with their personal information to register in the system. This information is used by mDurance to uniquely identify the specialist and also to preserve the patient's data collected by the system in a confidential and integral manner. As it mentioned in Section 3.3, some features

(a)



(b)



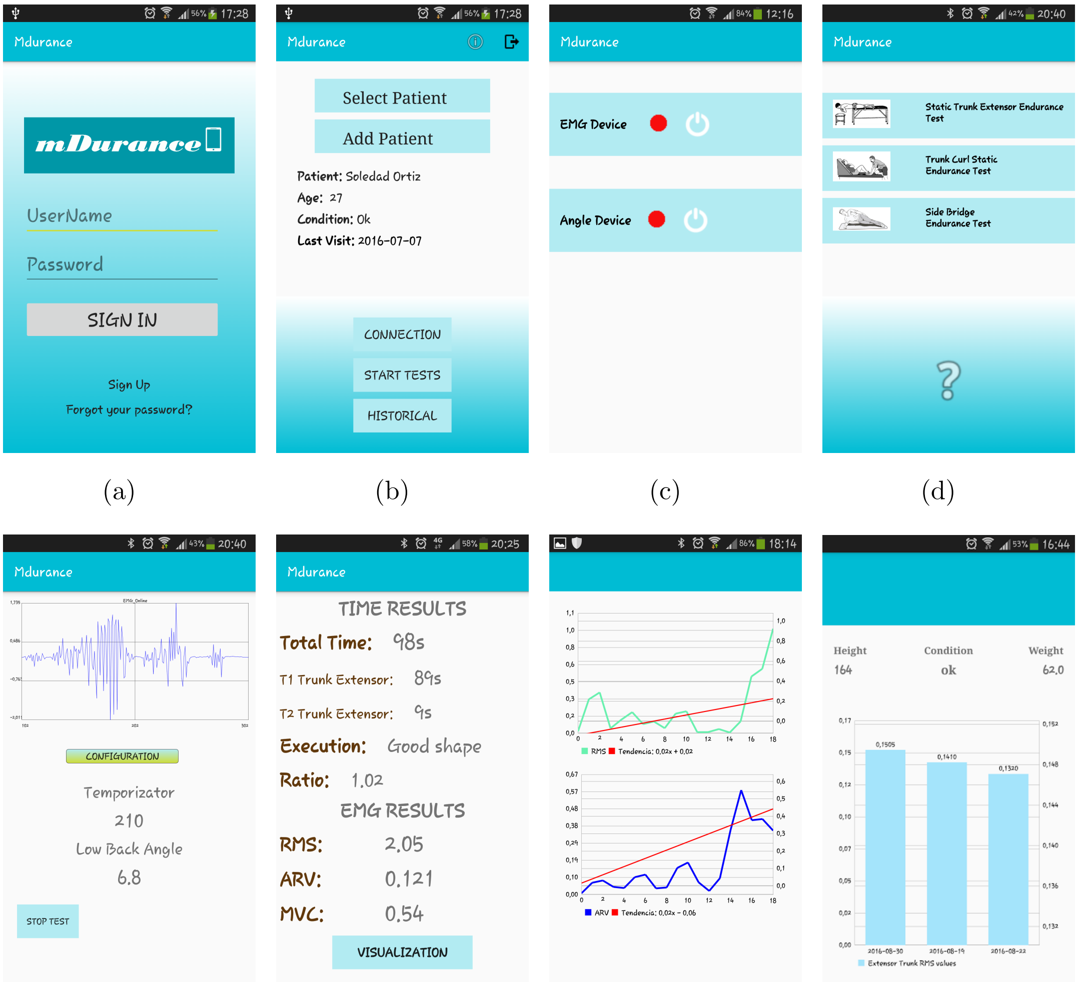
(c)

Figure 3.3: Sensor deployment for (**a**) STEET, (**b**) TCSET and (**c**) SBET procedures.

and resources available on mobile devices are not sufficient to provide the level of security and storage that users have come to expect. For improve the security, a new mechanism of authentication has been developed. It is treated of a scheme based on token. This authentication scheme uses a simple token-based HTTP Authentication scheme. When a new specialist is registered in the system, the server assigns him a new random password which is store in the server and is physically hidden (not visible to the possessor). However, it is transmitted for each authentication. For example, when a specialist wants to access to the application, the specialist introduces his username and his password. These are checked in the server and if they are right then the system sends his token corresponding. Token authentication is appropriate for client-server setups, such as native desktop and mobile clients. Respect to the mDurance storage functionality, it was built on top of the mHealthDroid Storage Manager (115), which provides a high level of abstraction from the underlying storage technology and enables data persistence, both locally and remotely. In the previous implementation, the mDurance app stored data locally on a SQLite database (116) deployed on the mobile phone SD card. If it is wanted to consult this, it can to be found in the Section 4.5.5 in (10). In this works, the mDurance storage is builds about a back-end web-service and stored data in the remotely in the cloud on a PostgreSQL database mentioned in Section 3.3.2.

Once an expert is logged into the application contents by their username and password (Figure 3.4(a)) the expert is directed to a new screen (Figure 3.4(b)), in which as first possibility, it can be either selected one of the existing patients in the system database associated to actual specialist or include a new one. Personal information, such as name, age, height, weight, gender and possible health conditions, are requested when filling in a new patient registry. Upon selecting a patient, their more relevant personal information is presented to the expert for quick inspection, including the date of the last endurance session and particular conditions from which they suffer. Moreover, from this main screen, the expert can either initiate the connection with the wearable sensors, start the endurance tests or visualize the historical data collected during previous sessions. The connection with the wearable sensors is performed by clicking on Connection (Figure 3.4(b)). During the very first configuration of the system, the sensors must be paired with the mobile device. To do so, it has been simplified the previous process described in the Section 4.4.3.1 in (10). Now, it is easier because the user just has to press the power button of each device and they are paired automatically. When this operation happens with success, the color of the status circles change from red to orange. After configuration, this one-time process is no longer required, unless the sensors are replaced. For to finalize the connection process, the user must to press the power button again. For to check that the process has finalized whit success, both status circles must be changed from orange to green. From this screen, the expert also can normally trigger the connection of the mobile and the wearable devices by checking the status circles.

Once the sensors are connected and in order to proceed with the execution of the tests, the expert has to press ”Start Tests” (Figure 3.4(b)). As a result,



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 3.4: mDurance application snapshots: (**a**) Login; (**b**) Specialist menu; (**c**) Sensors connection; (**d**) Endurance test selection; (**e**) Test execution; (**f**) Test results summary; (**g**) RMS and ARV results visualization along the time; and (**h**) Historical representation.

the specialist is directed to a new window in which the particular test to be performed can be chosen (Figure 3.4(d)). After selecting a test, another screen is displayed with the essential elements required by the expert to perform the test (Figure 3.4(e)). This includes a graph to visualize the recorded EMG signal at runtime, a timer to control the time left according to the maximum duration allowed for the realization of the test and the trunk angle continuously measured by the system. The trunk angle is particularly useful for the expert to determine when the patient is correctly positioned. Then, once the specialist determines that the starting position is reached, the test can be initiated by clicking on the corresponding button. The angle measured at that moment is saved as a reference and used by the system to check whether the user exceeds the range defined for each test as part of the termination criteria. Thus, if the patient relaxes their posture more than $\pm 10°$ in the STEET and SBET or $\pm 30°$ in the TCSET, the test is automatically finished. The end-of-test is also attained when it lasts more than 240 s or when the expert explicitly considers that it should be finalized, for which the stop button can be used. It is noteworthy that for each test realized, all the angles and EMG values measured during its assessment are stored in the database. Moreover, the new mechanism based on sliding window to measure the muscle fatigue is used in this section. This get EMG values during all the test, stores their in a virtual buffer and calculates the RMS and ARV value for each time window. These new RMS and ARV vectors are also stored in the database for their later use.

After the test finalization, the expert can observe a summary of the results (Figure 3.4(f)). This includes the total duration of the test (sum of the two attempts for the STEET case), the endurance ratio and the RMS, ARV and MVC values. The endurance ratio is a normalized value that represent the total time measured explained previously and divided by the theorist mean according to the healthy values that a healthy person must to obtain in the assessment of the corresponding test. The session is categorized into the results showed in Table 3.1.

	Bad result	**Good result**	**Perfect result**	**Execution ratio**
STEET	0 to 61s.	62 to 131s	132 to 240s	total_time/96.5
TCSET	0 to 133s	No specific	134 to 240s	total_time/134
SBET	0 to 49s	50 to 119s	120 to 240s	total_time/84.5

Table 3.1: Statistical overall duration of the patient and execution ratios.

In addition, in this work is presented a new graphic visualization of results

(Figure 3.4(g)). In this screen, it is showed the results obtained by the calculation of the RMS and ARV values through of sliding window mechanism explained previously. In this way, the expert can consult how a particular muscle has been behaved in function of the time and the test duration.

Finally, the expert can inspect the patient's historical data by clicking on the Historical button (Figure 3.4(b)). This opens a new screen, in which diverse types of representations can be selected, such as the time invested by the patient during the execution of the test and the muscle fatigue metrics. The results are depicted in a multi-date basis for the different past sessions registered in the system for the specific individual (Figure 3.4(h)).

4

Project Implementation

In this chapter, it is described how the new main modules mDurance explained in the previous chapter have been implemented. Along the Section 4.1 is explained the development of the back-end and of the API Rest service: the specialist application 4.1.1, the patient application 4.1.2, the results data application 4.1.3 and the tests data application 4.1.4. In addition, in this last section will be presented the configuration of PostgreSQL and Heroku with Django. Finally, in the last section 4.2 is developed the implementation of an API Restful with Android.

4.1 Back-end and API Rest service implementation

Designed to have a clear and concise syntax, Python is a language that enables developers to adopt several styles: object-oriented programming, functional programming and imperative programming. Actually, their demand has increased exponentially: from web development with Django to the financial industry and Big Data. The developed application represents the back-end of the mDurance. It consists of four main modules that comprehend all the features of the system: specialists, patients,tests data and results data. Each of one allows users to perform the following action depending of their need:

- Access the admin page/screen of the service.

- Insert a new specialist, patient, tests data or results data.
- List one or all the specialists, patients and data registered on the system
- Delete any existing specialist, patient, tests data or results data.
- Modify any existing specialist field, patient field, test data or results data field.
- Search any field for a specific patient, specialist or result.
- Order the patient or data by date, name, etc.

The first thing that it is need is to install Django and REST Framework in the system to start working. For this, it is used pip (Python Package Manager) from the computer console:

```
pip  install django djangorestframework
```

When it is installed, the next step is to create a Django project:

```
django-admin.py startproject mdurance
```

This creates a mdurance folder which will have another mdurance folder with files py that act like configuration files of the project. Django's philosophy is DRY (Do not Repeat Yourself), so the framework itself forces us to create applications so they can be reused. So in the mdurance project we will create all the applications mentioned in the previous paragraph: specialist, patients, tests and results data.

4.1.1 Specialists class

For that, it is accessed to the project folder and it is created the first application: specialist.

```
python manage.py startapp specialists
```

This will create a specialists folder with the py files of this application. This will create a specialists folder with the py files of this application. Django uses the MVC patron where the models are written in the model.py. However, in the case of the specialists, it is not necessary to create an specific model, since Django provides a class that contains a model by default to manage both the admins as the users with any privileges (in this case the specialists). Therefore, we take advantage of the *User* class of Django and there is not anything to implement in the model file.

The next step is to create a *serializer*, which translates among the model and the data that are send through any HTTP request of the API Rest. We create a serializer.py file in the specialists folder with the following content:

```
class SpecialistSerializer(serializers.Serializer):
    pk = serializers.ReadOnlyField()
    first_name = serializers.CharField()
    last_name = serializers.CharField()
    username = serializers.CharField()
    password = serializers.CharField()
    email = serializers.EmailField()

    def validate_username(self, username):
        existing = User.objects.filter(username=username).first()
        if existing:
            raise serializers.ValidationError("This username has
                already registered. Try it again")
        return username

    def validate_email(self, email):
        existing = User.objects.filter(email=email).first()
        if existing:
            raise serializers.ValidationError("This email has already
                registered. Try it again")
        return email

    def create(self, validated_data):
        instance = User()
        return self.update(instance, validated_data)
```

```
def update(self, instance, validated_data):
    instance.first_name = validated_data.get('first_name')
    instance.last_name = validated_data.get('last_name')
    instance.username = validated_data.get('username')
    instance.set_password(validated_data.get('password'))
    instance.email = validated_data.get('email')
    instance.save()
    return instance
```

At first, we have to indicate to the serializer the name of the fields that it has to serialize. Thus, the fields that the API Rest will send through HTTP requests are the primary key (pk), the first and second specialist's name, an username, a password and the specialist's email. The following two methods are used to validate both the username field as the email field for two specialist do not have the same username or email. If we want to be able to return complete object instances based on the validated data we need to implement one or both of the *create()* and *update()*. The method *create()* creates an instance of a specialist by *validated_data* that contains the deserialized values. Finally, in *update()*, calling *.save()* will either create a new instance, or update an existing instance, depending on if an existing instance was passed when instantiating the serializer class. It is noteworthy that we only have to declare to the Serializer which are the fields that it must represent and it is responsible for all. The next step is to implement the API, so we are going to detail the jobs that it does:

- In the URL /api/1.0/users/, is wanted to get a list with the specialists that there are in the system if somebody makes a GET request. If the request is POST, then a new specialist will be created.

- In the URL /api/1.0/users/ID/, is wanted to get the specialist's detail if somebody makes a GET request, while if the request is PUT, the specialist will be updated and if the request is DELETE will be removed of the system.

Therefore, we create an new api.py file in the specialists folder with the following content:

```
class SpecialistViewSet(ViewSet):

    permission_classes = (UserPermission,)

    def list(self, request):
        self.check_permissions(request)
        specialists = User.objects.all()
        paginator = PageNumberPagination()
        paginator.paginate_queryset(specialists, request)
        serializer = SpecialistSerializer(specialists, many=True)
        serialized_users = serializer.data
        return paginator.get_paginated_response(serialized_users)

    def create(self, request):
        self.check_permissions(request)
        serializer = SpecialistSerializer(data=request.data)
        if serializer.is_valid():
            new_user = serializer.save()
            return Response(serializer.data,
                status=status.HTTP_201_CREATED)
        else:
            return Response(serializer.errors,
                status=status.HTTP_400_BAD_REQUEST)

    def retrieve(self, request, pk):
        self.check_permissions(request)
        specialists = get_object_or_404(User, pk=pk)
        self.check_object_permissions(request, specialists)
        serializer = SpecialistSerializer(specialists)
        serialized_user = serializer.data
        return Response(serialized_user)

    def update(self, request, pk):
        self.check_permissions(request)
        specialists = get_object_or_404(User, pk=pk)
```

```
        self.check_object_permissions(request, specialists)
        serializer = SpecialistSerializer(instance=specialists,
            data=request.data)
        if serializer.is_valid():
            serializer.save()
            return Response(serializer.data, status=status.HTTP_200_OK)
        else:
            return Response(serializer.errors,
                status=status.HTTP_400_BAD_REQUEST)

    def destroy(self, request, pk):
        self.check_permissions(request)
        specialists = get_object_or_404(User, pk=pk)
        self.check_object_permissions(request, specialists)
        specialists.delete()
        return Response(status=status.HTTP_204_NO_CONTENT)
```

Django REST framework allows you to combine the logic for a set of related views in a single class, called a *ViewSet*. In other frameworks you may also find conceptually similar implementations named something like 'Resources' or 'Controllers'. A *ViewSet* class is simply a type of class-based View, that does not provide any method handlers such as *get()* or *post()*, and instead provides actions such as *list()* and *create()*. In addition, we have implemented three aditional methods: *retrieve()* get a specialist with a particular ID, *update()* updates any field of a particular specialist and *destroy()* removes a particular specialist through his ID. Again, just give some information to the class that we created it is enough: we say it where the information should take (queryset) and what to use as a translator (serializer_class). Django REST Framework are responsible of the rest.

Specialist's permissions

Another feature of Django REST framework are the permissions. One the one hand, REST framework includes a number of permission classes that we can use to restrict who can access a given view. On the other hand, Django allows personalize our own permissions. For the case of the specialists class, we are going to develop the permissions. In this case, we are looking for permissions which allows:

- Create a new specialist account to any people.
- List all the specialist only to the administrator/superuser.
- Retrieve to a particular specialist to the administrator/superuser.
- Delete to a particular specialist only the own specialist or the administrator/superuser.

We are going to need to create a custom permission.py file and fill it with the next code:

```
class UserPermission(BasePermission):

    def has_permission(self, request, view):
        if view.action == "create":
            return True
        elif request.user.is_superuser:
            return True
        elif view.action in ['retrieve', 'update', 'destroy']:
            return True
        else:
            return False

    def has_object_permission(self, request, view, obj):
        return request.user.is_superuser or request.user == obj
```

Through *has_permission* is defined if the authenticated user in a particular request has permission to realize the GET, POST, PUT or DELETE action indicated in the request. The methods should return True if the request should be granted access, and False otherwise. However, this method is implemented to treat the actions executed about the set of the specialists. The instance-level *has_object_permission* method will only be called if the view-level *has_permission* checks have already passed and it is in charge of allowing perform actions on a particular specialist (GET, PUT or DELETE).

Authentication

Such as we mentioned in the Section 3.4, for improve the security, we have developed a new mechanism of authentication. It is treated of a scheme based on token. This authentication scheme uses a simple token-based HTTP Authentication scheme. When a new specialist is registered in the system, the server assigns him a new random password which is store in the server and is physically hidden. Thus, if we want that every specialists to have an automatically generated Token when they are registered, we can simply catch the User's *post_save* signal introducing the next code in the model.py file:

```
@receiver(post_save, sender=settings.AUTH_USER_MODEL)
def create_auth_token(sender, instance=None, created=False, **kwargs):
    if created:
        Token.objects.create(user=instance)
```

Now, if successfully authenticated, *TokenAuthentication* class provides the following credentials:

```
Authorization: Token 9944b09199c62bcf9418ad846dd0e4bbdfc6ee4b
```

From now on, for specialist to authenticate, the token key should be included in the Authorization HTTP header. The key should be prefixed by the string literal "Token", with whitespace separating the two strings such as we mentioned previously.

Creating the api URL

Finally we have to connect this *ViewSet* and the *TokenAuthentication* with the API's URLs that we mentioned previously. For this, we use a *Router*. We open the urls.py file that is included in the specialists folder and we add these lines:

```
router = DefaultRouter()
router.register(r'users', SpecialistViewSet, base_name='specialist')
urlpatterns = [
    url(r'1.0/', include(router.urls))
```

```
]
```

A *Router* allows the urlconf to be automatically generated. With this, we are allowing to can be sent requests to the URL 1.0/users/. At end, we have to join the specific urls.py file that we have created in the specialist folder with the general urls.py file created in the mdurance project folder that must contains all the available urls of the project. For this, we have to add in this file:

```
urlpatterns = [
    url(r'^admin/', admin.site.urls),
    url(r'api/', include(specialists_api_urls)),
    url(r'^api-token-auth/\$, views.obtain_auth_token),
```

In addition, when using TokenAuthentication, it may be wanted to provide a mechanism for clients to obtain a token given the username and password. REST framework provides a built-in view to provide this behavior. To use it, we added the *obtain_auth_token* view in the urls.py file. In Figure 4.1 is showed the final result for this application:

Figure 4.1: GET requests: complete specialists list.

4.1.2 Patients class

Now, we are going to create another application in the project. This new application will manage all related with the patients:

```
python manage.py startapp patients
```

This creates a patients folder with the py files of this application. In the case of the patients, it is necessary to create an specific model to specific the fields that we will want to create in the database. Then, we add the next code in the model.py file:

```
MALE = 'MALE'
FEMALE = 'FEMALE'
GENDER = (
(MALE, 'Male'),
(FEMALE, 'Female')
)
class Patient(models.Model):
    specialist = models.ForeignKey(User)
    dni = models.CharField(max_length=36, unique=True)
    name = models.CharField(max_length=100)
    age = models.IntegerField()
    sex = models.CharField(max_length = 6, choices = GENDER)
    height = models.IntegerField()
    weight = models.FloatField()
    email = models.EmailField(max_length=100, unique=True)
    condition = models.TextField()
    created_at = models.DateTimeField(auto_now_add=True)
    modified_at = models.DateTimeField(auto_now=True)
```

Due to all the patients will have an associated specialist, it will be easier if we can create a recursive relationship among both. To create a recursive relationship (an object that has a many-to-one relationship with itself) with the specialist, Django provides *models.ForeignKey()*. Finally, if we want to relate it with the specialists class (which was associated with the *User* class of Django), we add the User parameter into the bracket. The dni, name, age, sex, height, weight, email and condition fields correspond with the basic information of the patients.

Finally, created_at and modified_at are two *DateTime* Django fields that indicate when a patient has been created or modified automatically. In addition, Django checks that both the dni as the email are uniques when a new patient is created. In definitive, Django provides an easy way to manage the database and it safes us a lot of time. Before to check it, we have to create the migration files for to create our database schema posteriorly (by default, Django uses SQLite but we mentioned in the previous chapter that we use PostgreSQL). We return to our console and execute:

```
python manage.py makemigrations
```

This creates an 000_ initial.py file in the migrations folder of our patient folder. These migrations are created each time that there is any change in the models (changes in structures of SQL tables) or when we created them by first time. Now, we have to apply the migration with the migrations scripts created:

```
python manage.py migrate
```

The next step is to create a *serializer* again. We create a serializer.py file in the patients folder with the following content:

```
class PatientSerializer(serializers.ModelSerializer):

    class Meta:
        model = Patient
        read_only_fields = ('specialist',)


class PatientListSerializer(PatientSerializer):

    class Meta(PatientSerializer.Meta):
        fields = ('id', 'name', 'age','dni', 'sex'
                  , 'created_at', 'email', 'height'
                  , 'weight', 'condition')
```

We just indicate to the serializer which is the model that it must represent and it handles everything. With this, the serializer already knows that when it receives an object, it must serialize it with the patient's model fields. The *Mod-*

elSerializer class provides a shortcut that lets us automatically create a *Serializer* class with fields that correspond to the Model fields. The *ModelSerializer* class is the same as a regular *Serializer* class, except that: it will automatically generate a set of fields for us, based on the model; It will automatically generate validators for the serializer; It includes simple default implementations of *create()* and *update()*. Otherwise, if we want to implement another serializer to can list the patients, we have to create another serializer that inherits of the first serializer (PatientSerializer). To specify which fields to include when we want to list the patients, it is strongly recommended that we explicitly set all fields that should be serialized using the fields attribute. This will make it less likely to result in unintentionally exposing data when our models change. The next step is to implement the API, so we are going to detail the jobs that it does:

- In the URL /api/1.0/patients/, is wanted to get a list with the patients that there are in a particular specialist's account if this specialist makes a GET request. In the case that an admin/superuser realizes the GET request, all the patients in the system will be show. In the case that the request is POST, then a new patient will be created. In addition, it will be related with the specialist who has added to it automatically.

- In the URL /api/1.0/patients/ID/, is wanted to get the patient's detail if somebody makes a GET request, while if the request is PUT, the patient will be updated and if the request is DELETE will be removed of the system.

Therefore, we create an new api.py file in the patients folder with the following content:

```
class PatientViewSet(ModelViewSet):
    permission_classes = (IsAuthenticated,)
    queryset = Patient.objects.all()
    filter_backends = (SearchFilter, OrderingFilter)
    search_fields = ('specialist', 'dni', 'email')
    ordering_fields = ('id', 'created_at')

    def get_queryset(self):
```

```
        if self.request.user.is_superuser:
            patients = Patient.objects.all()
        else:
            patients =
                Patient.objects.filter(specialist=self.request.user)
        return patients

    def get_serializer_class(self):
        if self.action == 'list':
            return PatientListSerializer
        else:
            return PatientSerializer

    def perform_create(self, serializer):
        serializer.save(specialist=self.request.user)
```

Again, we just give a bit of information to Django is sufficient: we say it where to get the patient's information (queryset) and what it must use as translator (serializer_class). To finalize the recursive relationship that we mentioned previously, we have added the method *perform_create*, which assigns the authenticated specialist to the new patient automatically when this is going to be created. We have added another extra code to implement all the features necessaries for the commodity of the specialists by filter search and filter order.

Patients's permissions

REST framework includes a number of permission classes that we can use to restrict who can access a given view. In this case, we are looking for permissions which allows:

- Create a new patient's account to a particular specialist.
- List all the patients only to the administrator/superuser.
- List all the patients of a particular specialist's account to the specialist who is authenticated in the system.
- Retrieve to a particular patient to the administrator/superuser or to the specialist who is authenticated in the system.

- Delete to a particular patient only the own specialist or the administrator/-superuser.

The *IsAuthenticated* permission class will deny permission to any unauthenticated specialist, and allow permission otherwise. In addition, we develop the method *get_queryset()* which filters the corresponding patients for each specialists or for the admin/superuser.

Creating the api URL

Finally we have to connect the *ModelViewSet* with the API's URLs that we mentioned previously. For this, we use again a *Router*. We open the urls.py file that is included in the patients folder and we add the next lines:

```
router = DefaultRouter()
router.register(r'patients', PatientViewSet)
urlpatterns = [
    url(r'1.0/', include(router.urls))
]
```

Finally, we have to join the specific urls.py file that we have created in the patients folder with the general urls.py file created in the mdurance project folder that must contains all the available urls of the project. For this, we have to add in this file the new url:

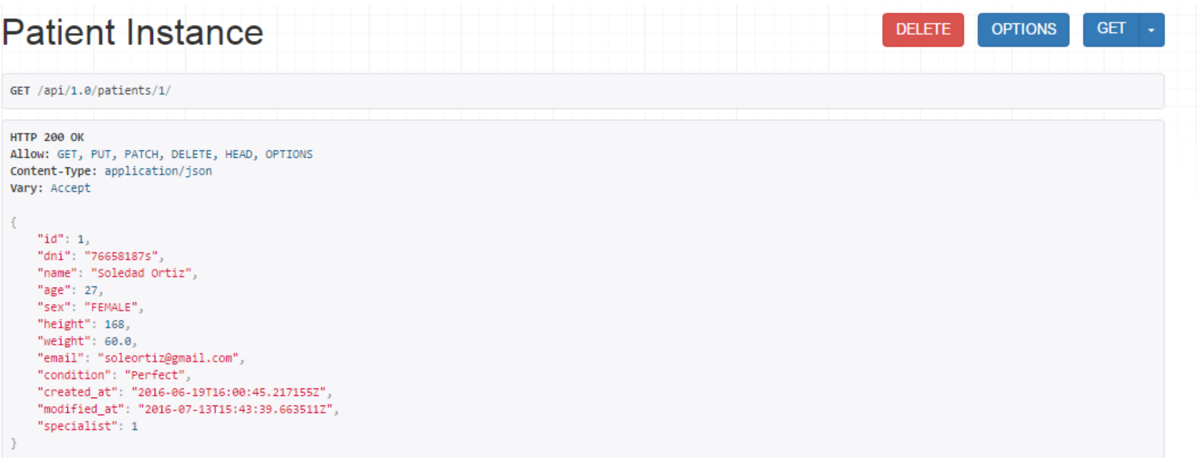
```
urlpatterns = [
    url(r'^admin/', admin.site.urls),
      url(r'^api-token-auth/\$, views.obtain_auth_token),
    url(r'api/', include(specialists_api_urls)),
      url(r'api/', include(patients_api_urls)),
```

In Figure 4.2 is showed the final result for this application:

4.1.3 Test results class

In this section, we are going to create the corresponding application to save the test results summarized in Figure 3.4(f). However, the process of creation is very similar to the process of creation of the patient class. Thus, we are just going

(a)



(b)

Figure 4.2: GET requests: (**a**) Complete patient list; (**b**) Detail view of a patient.

to explain the model of one test (STEET) for to avoid duplicate code. The next code can be used to replicate the rest of the tests (TCSET, SBETR and SBETL):

```
python manage.py startapp steet_results
```

This creates a steet_results folder with the py files of this application. Like the case of the patients, it is necessary to create an specific model to specific the fields that we will want to create in the database. Then, we add the next code in the model.py file:

```
class STEETResult(models.Model):
    patient = models.ForeignKey(Patient)
    specialist = models.ForeignKey(User)
    created_at = models.DateTimeField(auto_now_add=True)

    calibration_angle = models.FloatField(blank=True, null=True)
    total_time = models.IntegerField(blank=True, null=True)
    first_time = models.IntegerField(blank=True, null=True)
    second_time = models.IntegerField(blank=True, null=True)
    RMS = models.FloatField(blank=True, null=True)
    ARV = models.FloatField(blank=True, null=True)
    MVC = models.FloatField(blank=True, null=True)
    age_patient = models.IntegerField()
    height_patient = models.IntegerField()
    weight_patient = models.FloatField()
    condition_patient = models.CharField(max_length=50)
```

Due to that all the results will have an associated patient and this will have an associated specialist, it will be easier if we can create a recursive relationship among all. To create a recursive relationship we use again *models.ForeignKey()*. The calibration_angle, total, first, second time, RMS, ARV and MVC fields correspond with the results information of the tests. The age, height, weight and condition fields correspond with the patient information which we are considering relevant to influence the test results. Finally, created_at indicates when a patient has done this test. Now, we have just to replicate all the steps followed in the previous patients section: to make the migrations, to create the serializer by *ModelSerializer*, to implement the api by *ModelViewSet* with the corresponding permissions class (*IsAuthenticated* and to join the new URLs with the general urls.py file. All the new urls created for all the tests are:

- In the URL /api/1.0/sorensenTest/, is wanted to get a list with the results associated to the STEET test that the patients have gotten for a particular specialist's account.

- In the URL /api/1.0/flexorTest/, is wanted to get a list with the results associated to the TCSET test that the patients have gotten for a particular specialist's account.
- In the URL /api/1.0/rightLateralTest/, is wanted to get a list with the results associated to the SBETR test that the patients have gotten for a particular specialist's account.
- In the URL /api/1.0/leftLateralTest/, is wanted to get a list with the results associated to the SBETL test that the patients have gotten for a particular specialist's account.

Thus, the urls.py files would be:

```
urlpatterns = [
    url(r'^admin/', admin.site.urls),
      url(r'^api-token-auth/\$, views.obtain_auth_token),
    url(r'api/', include(specialists_api_urls)),
      url(r'api/', include(patients_api_urls)),
      url(r'api/', include(STEET_api_urls)),
      url(r'api/', include(TCSET_api_urls)),
      url(r'api/', include(SBETR_api_urls)),
      url(r'api/', include(SBETL_api_urls)),
```

In Figure 4.3 is showed the final result for the STEET application:

4.1.4 Test data class. Implementing PostgreSQL and Heroku

Only it remains for us to describe the application which stores all the data that the sensors stream during the tests. The sampling rate of each sensors are 15 Hz for the IMU and 10 Hz for the EMG. In one second each sensor will stream 10 and 15 data. Thus, the system must be prepare to store a maximum of 3600 data for the IMU and 2400 data for the EMG. The Django database by default is SQLite and such as we had implemented in the past project (10), for to can be store this quantity of data, we will have to store them registering a ID for each one of them.

```
HTTP 200 OK
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "count": 46,
    "next": null,
    "previous": null,
    "results": [
        {
            "id": 103,
            "patient": {
                "id": 25,
                "dni": "3542",
                "name": "Zambrano",
                "age": 25,
                "sex": "MALE",
                "height": 172,
                "weight": 81.0,
                "email": "zambrano@guadix.com",
                "condition": "ok",
                "created_at": "2016-08-26T18:22:25.357508Z",
                "modified_at": "2016-08-26T18:22:25.357553Z",
                "specialist": 11
            },
            "specialist": 11,
            "created_at": "2016-08-31T20:04:17.648809Z",
            "total_time_sorensen": 132,
            "RMS_sorensen": 0.105458991243086,
            "ARV_sorensen": 0.0831699141750926,
            "first_time_sorensen": 74,
            "second_time_sorensen": 58,
            "MVC_sorensen": 0.365323573350906,
            "calibration_sorensen_angle": 19.3938808441162,
            "age_patient": 25,
            "height_patient": 172,
            "weight_patient": 81.0,
            "condition_patient": "ok"
        },
```


Figure 4.3: GET requests: complete STEET results list.

This is an ineffective method, so we decided to look for more options to solve this problem. In this context appears PostgreSQL. This database supports an array datatype. This is most similar to arrays in many statically-typed languages such as C or Java, in that we explicitly declare that we want an array of a specific type (for instance, an array of integers, floats or an array of strings). Thus, if we would want to use these fields, we had to migrate from SQLite to PostgreSQL.

Configuring PostgreSQL and Heroku with Django

We are going to configure PostgreSQL (version 9.5) in Django and to deploy a back-end service in Heroku. Heroku is a Platform-as-a-Service (PaaS) of cloud computing that allows us to deploy our application in the cloud for free. The first step is to create an Heroku account and to create into the platform an PostgreSQL database.

To register the change into the our application, it is necessary to explain what is the settings.py file. A Django settings file contains all the configuration of your

Django installation: libraries, modules and applications registered in the system, the authentication class which is used in the applications (in our case, token based) and between another feature, a dictionary containing the settings for all databases to be used with Django. The simplest possible settings file is for a single-database setup using SQLite. This can be configured using the following:

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': 'mydatabase',
    }
}
```

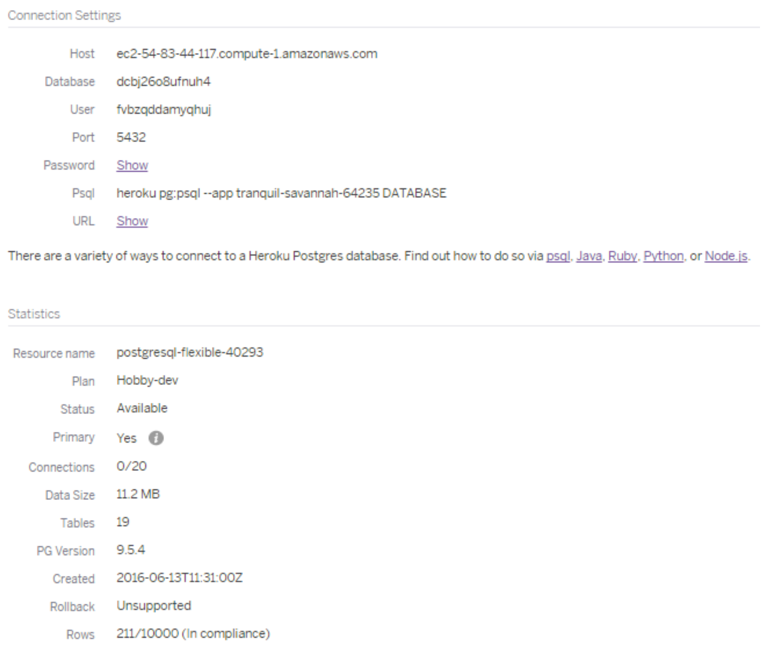


Figure 4.4: The figure shows the data and features of the PostgreSQL database created into Heroku.

However, when connecting to other database backends, such as MySQL, Oracle, or PostgreSQL, additional connection parameters will be required. To connect our PostreSQL database, it is necessary to see the basic information about the database in Figure 4.4 and to fill in the next parameters:

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql_psycopg2',
        'NAME': 'dcbj26o8ufnuh4',
        'USER': 'fvbzqddamyqhuj',
        'PASSWORD': 'glmyLQwyXiHhT-9eO6L-ZzTyoe',
        'HOST': 'ec2-54-83-44-117.compute-1.amazonaws.com',
        'PORT': '5432',
    }
}
```

Installing the configuration tools

Previously to configure Heroku, we have to install the Heroku toolbelt (v3.3.0) tool. This tool is formed of three basic utilities:

- Heroku client: CLI tool for the creation and management of Heroku applications.
- Foreman: it is used to execute applications locally.
- Git: it is a version control system that we will use in the applications hosted in Heroku.

Configuring Procfile, requirements and wsgi

Now, we have to use a Procfile file. This file must be defined in our root folder (mdurance). If we want that our application is executed, we must define a dyno. A dyno is a container, which executes the command that we specify. We have also to run Gunicorn for to can execute the application in Heroku. Thus, we create the Procfile file and add the next line:

```
web: gunicorn mdurance.wsgi
```

Heroku recognizes a Python application by the existence of a requirements.txt file in the root directory. In this file are specified the additional Python modules that the application needs. For the right deploy in the application, we have to have defined the next modules:

```
dj-database-url==0.4.1
dj-static==0.0.6
```

```
Django==1.9.6
django-allauth==0.25.2
django-rest-auth==0.7.0
django-toolbelt==0.0.1
djangorestframework==3.3.3
gunicorn==19.6.0
psycopg2==2.6.1
python-openid==2.2.5
requests==2.10.0
six==1.10.0
static3==0.7.0
wheel==0.24.0
```

Then, we modify the wsgi.py file adding the next code:

```
from django.core.wsgi import get_wsgi_application
from dj_static import Cling

os.environ.setdefault("DJANGO_SETTINGS_MODULE", "mdurance.settings")
application = Cling(get_wsgi_application())
```

With these steps, we already have the application configured to be deployed in Heroku.

Creating a Git repository for the application

The next step is to create a repository in git, where we will store the application. The first step is to define a .gitignore hidden file where they are defined the files and folders that we want to be ignored by our repository. The .gitignore file must contain the next restrictions:

```
venv
*.pyc
staticfiles
```

Then, we download the git console and into the our root directory, we save the changes with the next commands:

```
git init
```

```
git add .
git commit -m 'initial commit'
```

Uploading the Heroku application

Finally, the last step is to upload the application that we have just to create to a Heroku repository. For this work, we use the next lines:

```
heroku create
git push heroku master
heroku ps:scale web=1
git push heroku master
```

The first command has created a repository in Heroku where we upload the application with the second command. The last line is used to run the application that is already installed in Heroku. With this, we have already finished of the basic Heroku configuration.

Creating the test data application

To end, we only remains for us to create the Django application such as we described in the previous sections (specialists, patients and results data). The unique change between results data and this application would be the models. In this case, we will use the ArrayField field that we explained previously in the start of this section. Thus, the model.py file for this application is:

```
class TestData(models.Model):
    specialist = models.ForeignKey(User)
    patient = models.ForeignKey(Patient)
    label = models.CharField(max_length=36)
    angle_data = ArrayField(base_field=models.FloatField())
    emg_data = ArrayField(base_field=models.FloatField())
    rms_data = ArrayField(base_field=models.FloatField())
    arv_data = ArrayField(base_field=models.FloatField())
    created_at = models.DateTimeField(auto_now_add=True)
```

Due to that all the results will have an associated patient and this will have an associated specialist, we have to create again a recursive relationship among all. The angle_data and emg_data corresponds to all the angle and EMG data measured during the test. Both rms_data as arv_data are referred to the RMS and ARV values calculated by the time windows methods that we explained in previous chapter. Finally, created_at indicates when has been created each entry. Now, we have just to replicate all the steps followed in the previous results data section: to make the migrations, to create the serializer by *ModelSerializer*, to implement the api by *ModelViewSet* with the corresponding permissions class (*IsAuthenticated* and to join the new URLs with the general urls.py file. The new urls created for the requests are:

- In the URL /api/1.0/dataTest/, is wanted to get a list with the tests data associated to the all testes that the patients have gotten for a particular specialist's account.
- In the URL /api/1.0/dataTest/ID, is wanted to get a list with the tests data associated to a particular ID.

In Figure 4.5 is showed the final result for the STEET application:

4.2 Implementing an API Restful with Android

How it was commented in the previous section, working with REST services, the requests to the service will not be done through unique URL, if not it will be determined the action to realize according the URL accessed and the HTTP action used to realize the request (GET, POST, PUT or DELETE). In the next section, they are explained the implementation and integration of this methods in the Android application.

4.2.1 Insert specialists, patients, results and tests data

Such as it was commented in the past section, the new specialists and patients insertion will be realized in the next URLs respectively:

```
https://tranquil-savannah-64235.herokuapp.com/api/1.0/users/
https://tranquil-savannah-64235.herokuapp.com/api/1.0/patients/
https://tranquil-savannah-64235.herokuapp.com/api/1.0/sorensenTest/
```

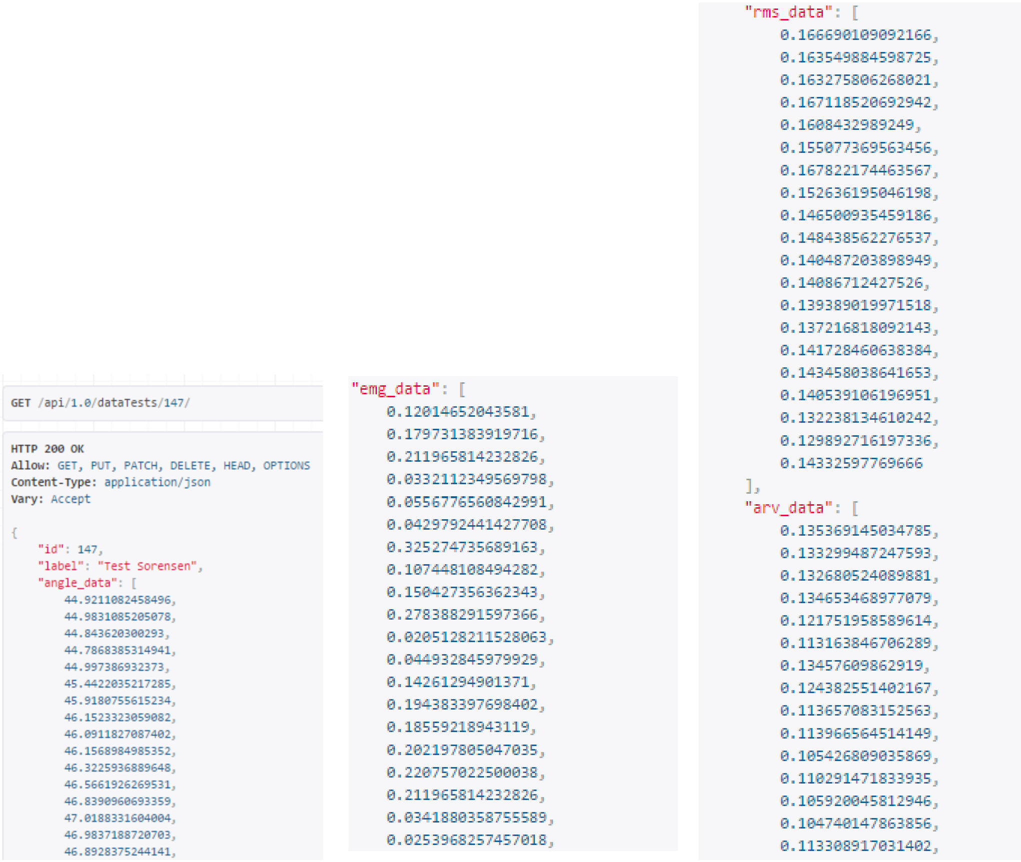


Figure 4.5: GET requests to a particular ID test data: (**a**) angle_data values; (**b**) emg_data values; (**c**) rms_data and arv_data values.

```
https://tranquil-savannah-64235.herokuapp.com/api/1.0/dataTest/
```

We will use the POST method and we will have to include in the request a JSON object, which must contain the new specialist, patient, results or tests data that were mentioned in the subsection 4.1.1, subsection 4.1.2, subsection 4.1.3, subsection 4.1.4 respectively. It is noteworthy that the ID fields that are calculated automatically and we have not to include them in the request. To achieve this, we have to start creating a new *HttpClient* object, which is the attendant of to realize the HTTP communication with the server by the provided data previously. After this, we create the POST request creating a new *HttpPost* object and indicating the URL to the service. We modify the HTTP content-type attribute by *setHeader()* method to indicate that the data format that we will use in the communication. Moreover, we have to indicate that the communication will be JSON, which MIME-Type is "'application/json"'. The difference between to insert a specialist or a patient is that for to insert the second, we also need the specialist token. In the next section will be explained how to get it. However, we can mention that this parameter will be included in the POST request by the *setHeader()* method such as it is showed in the code:

```
% Specialist Header insertion
HttpClient httpClient = new DefaultHttpClient();
HttpPost httpPost = new
   HttpPost("https://tranquil-savannah-64235.herokuapp.com/api/1.0/users/");
httpPost.setHeader("Content-Type", "application/json");

% Patient Header insertion
HttpClient httpClient = new DefaultHttpClient();
HttpPost httpPost = new
   HttpPost("https://tranquil-savannah-64235.herokuapp.com/api/1.0/patients/");
httpGet.setHeader("Authorization", token);
httpPost.setHeader("Content-Type", "application/json");

% Results data Header insertion
HttpClient httpClient = new DefaultHttpClient();
HttpPost httpPost = new
   HttpPost("https://tranquil-savannah-64235.herokuapp.com/api/1.0/sorensenTest/");
httpGet.setHeader("Authorization", token);
httpPost.setHeader("Content-Type", "application/json");

% Test data Header insertion
HttpClient httpClient = new DefaultHttpClient();
HttpPost httpPost = new
   HttpPost("https://tranquil-savannah-64235.herokuapp.com/api/1.0/dataTest/");
httpGet.setHeader("Authorization", token);
httpPost.setHeader("Content-Type", "application/json");
```

The next step will be to create the JSON object and to include it with the request. The request must contain the new specialist and patient, results or tests data. For this, we create a new *JSONObject* and add the necessary attributes with their respective values by the *put()* method. Finally, we associate this JSON object to the HTTP request and we convert it to *StringEntity* and include it in the request by *setEntity()*. In addition, in the tests data is neccesary to use the *accumulate()* method. This method appends the tests values to the array already mapped to each field (angle_data, emg_data, rms_data and arv_data).

```
% Specialist JSON OBJECT
JSONObject jsonObject = new JSONObject();
jsonObject.put("first_name", mSpecialist.getName());
jsonObject.put("last_name", mSpecialist.getLast_name());
jsonObject.put("username", mSpecialist.getUsername());
jsonObject.put("password", mSpecialist.getPassword());
jsonObject.put("email", mSpecialist.getEmail());

StringEntity stringEntity = new StringEntity(jsonObject.toString());
httpPost.setEntity(stringEntity);

% Patient JSON OBJECT
JSONObject jsonObject = new JSONObject();
jsonObject.put("name", mPatients.getName());
jsonObject.put("dni", mPatients.getDni());
jsonObject.put("age", mPatients.getAge());
jsonObject.put("sex", mPatients.getGender());
jsonObject.put("height", mPatients.getHeight());
jsonObject.put("weight", mPatients.getWeight());
jsonObject.put("email", mPatients.getEmail());
jsonObject.put("condition", mPatients.getCondition());

% Results data JSON OBJECT
JSONObject jsonObject = new JSONObject();
jsonObject.put("patient", Integer.parseInt(pk));
jsonObject.put("age_patient", age);
jsonObject.put("height_patient", height);
```

```
jsonObject.put("weight_patient", weight);
jsonObject.put("condition_patient", condition);
jsonObject.put("calibration_sorensen_angle", mPatient.getCalAngle());
jsonObject.put("total_time_sorensen", mPatient.gettTime());
jsonObject.put("first_time_sorensen", mPatient.getfTime());
jsonObject.put("second_time_sorensen", mPatient.getsTime());
jsonObject.put("RMS_sorensen", mPatient.getRmsEMG());
jsonObject.put("ARV_sorensen", mPatient.getArvEMG());
jsonObject.put("MVC_sorensen", mPatient.getMVC());

% Tests data JSON OBJECT
JSONObject jsonObject = new JSONObject();
jsonObject.put("patient", pk);
jsonObject.put("label", "Test Sorensen");
for (int i=0; i<bdAngles.size(); i++)
      jsonObject.accumulate("angle_data", bdAngles.get(i));
for (int j=0; j<bdEMG.size(); j++)
      jsonObject.accumulate("emg_data", bdEMG.get(j));
for (int i=0; i<bdRMSWindow.size(); i++)
      jsonObject.accumulate("rms_data", bdRMSWindow.get(i));
for (int i=0; i<bdARVWindow.size(); i++)
      jsonObject.accumulate("arv_data", bdARVWindow.get(i));
```

Once the request has been created, we have to realize the call to the service by the *execute()* method of the HttpClient object and to get the result by *getEntity()*. This result is got with the form of HttpEntity object, but we can convert it in a String by the static *EntityUtils.toString()* method.

```
HttpResponse httpResponse = httpClient.execute(httpPost);
HttpEntity entity = httpResponse.getEntity();
String responseBody = EntityUtils.toString(entity);
```

4.2.2 Specialist login

To perform the login action, Django provides us of a URL where to make the login request for to get the token corresponding to the specialist who runs the login request. Django provides the next URL:

`https://tranquil-savannah-64235.herokuapp.com/api-token-auth/`

We will use the POST method and we will have to include in the request a JSON object, which must contain the username and password fields (obtained by the login form of the figure 3.4(a)) to check them in the server. If both the username as the password are right, the server will return us the token corresponding to the specialist who is making the request by the *HttpEntity* object. Once we get the token, we store it in the system preferences as we use it for the rest of requests from now.

```
HttpClient httpClient = new DefaultHttpClient();
HttpPost httpPost = new
    HttpPost("https://tranquil-savannah-64235.herokuapp.com/api-token-auth/");
httpPost.setHeader("Content-Type", "application/json");
JSONObject jsonObject = new JSONObject();

jsonObject.put("username", mSpecialist.getUsername());
jsonObject.put("password", mSpecialist.getPassword());
StringEntity stringEntity = new StringEntity(jsonObject.toString());
httpPost.setEntity(stringEntity);

HttpResponse httpResponse = httpClient.execute(httpPost);
int statusCode = httpResponse.getStatusLine().getStatusCode();
if (statusCode == 200) {
   HttpEntity entity = httpResponse.getEntity();
   String responseBody = EntityUtils.toString(entity);
   token = "Token " + responseBody.substring(10, 50);
   savePreferences();
   return true;
} else {
   return false;
}
```

4.2.3 Update an existent patient

The URL used for the patients updating will be the same that the previous URL for to insert patient but adding the ID of the patient who wants to update:

```
https://tranquil-savannah-64235.herokuapp.com/api/1.0/patients/ID/
```

In this case, the JSON object sent as entry must contain again the new values to update. To update the patient, we use the PATCH request (*HttpPatch* object). It is noteworthy that the difference between PUT and PATCH request is that if we use PUT, we should update all the patient fields and if we use PATCH, we can update the patient field that we want. It should be pointed that we have to add the specialist token obtained in the previous section to can get permission to update the patient.

```
HttpClient httpClient = new DefaultHttpClient();
HttpPatch httpPatch = new HttpPatch(
           "https://tranquil-savannah-64235.herokuapp.com/api/1.0/patients/"
              + key + "/");
httpPatch.setHeader("Content-Type", "application/json");
httpPatch.setHeader("Authorization", token);
JSONObject jsonObject = new JSONObject();

Boolean result = false;

if (mName != null)
     jsonObject.put("name", mName.trim());
if (mAge != 0)
     jsonObject.put("age", mAge);
if (mEmail != null)
     jsonObject.put("email", mEmail.trim());
if (mHeight != 0)
     jsonObject.put("height", mHeight);
if (mWeight != 0)
     jsonObject.put("weight", mWeight);
if (mCondition != null)
```

```
    jsonObject.put("condition", mCondition.trim());

StringEntity stringEntity = new StringEntity(jsonObject.toString());
httpPatch.setEntity(stringEntity);

HttpResponse httpResponse = httpClient.execute(httpPatch);
statusCode = httpResponse.getStatusLine().getStatusCode();
if (statusCode == 200) {
    result = true;
}
```

In the next image 4.6, it is showed the options to update or delete any patient:

Figure 4.6: Patient information screen, where a specialist can update or delete any registered patient in the system.

4.2.4 Delete existent patients

The patient elimination is realized through the next URL:

https://tranquil-savannah-64235.herokuapp.com/api/1.0/patients/ID/

The ID is again the patient ID to eliminate. In addition, we use the DELETE request (*HttpDelete* object) to identify the action that we want to realize. In this case, it is not necessary to give any object of entry with the request. Thus, the code is simpler than previous:

```
HttpClient httpClient = new DefaultHttpClient();
HttpDelete httpDelete = new HttpDelete(
           "https://tranquil-savannah-64235.herokuapp.com/api/1.0/patients/"
              + key + "/");
httpDelete.setHeader("Content-Type", "application/json");
httpDelete.setHeader("Authorization", token);
Boolean result = false;

HttpResponse httpResponse = httpClient.execute(httpDelete);
statusCode = httpResponse.getStatusLine().getStatusCode();
if (statusCode == 200) {
     result = true;
}
```

In the previous image 4.7, it is showed the screen where a specialist can delete any patient when presses the deleted button.

4.2.5 List patients

This action is a little different to the previous. In this case, the result returned by the service will be an JSON object and not a simple value as in the previous cases. The URL used for this request is:
`https://tranquil-savannah-64235.herokuapp.com/api/1.0/patients/`

In this case, we use a GET request (*HttpGet* object). The way of to realize the request is analogous to the previous. Where the differences appear will be when we will want to treat the result returned by the service, after to call to

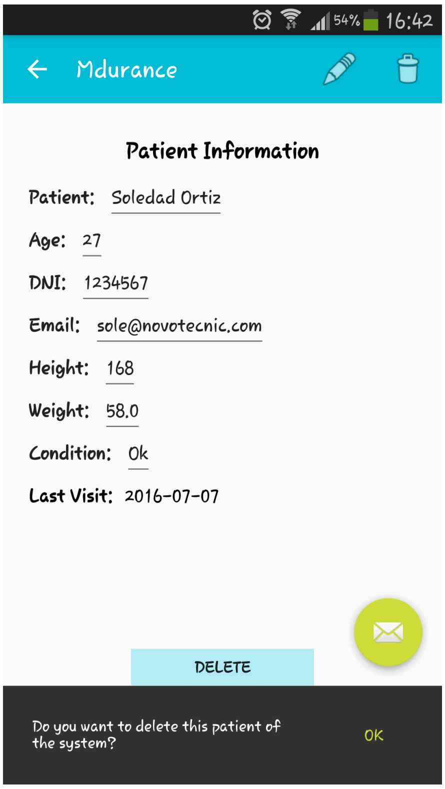


Figure 4.7: Patient information screen, where a specialist can update or delete any registered patient in the system.

the *getEntity()* method. To treat it, we create a new JSONArray object due to we received an elements array. Then, we can access to each one of them by the *getJSONObject()*, to the which we will go giving it the index of each element. For to know how many elements the array contains, we can use the *length()* method of the JSONArray object. Then, we can access to the object attributes using the *get()* methods according the type of each attribute (*getInt()*, *getString()*, etc).

```
HttpClient httpClient = new DefaultHttpClient();
HttpGet httpGet = new
    HttpGet("https://tranquil-savannah-64235.herokuapp.com/api/1.0/patients/");
httpGet.setHeader("Authorization", token);
httpGet.setHeader("Content-Type", "application/json");

HttpResponse response = httpClient.execute(httpGet);
```

```
String respBody = EntityUtils.toString(response.getEntity());

JSONObject jsonObject = new JSONObject(respBody);
JSONArray jsonArray = jsonObject.getJSONArray("results");
for (int i = 0; i < jsonArray.length(); i++) {
JSONObject object = jsonArray.getJSONObject(i);
String patient_id = Integer.toString(object.getInt("id"));
String patient_name = object.getString("name");
String patient_dni = object.getString("dni");
int patient_age = object.getInt("age");
String patient_gender = object.getString("sex");
String patient_created = object.getString("created_at");
patient_created = patient_created.substring(0, 10);
String patient_email = object.getString("email");
int patient_height = object.getInt("height");
double patient_weight = object.getDouble("weight");
String condition = object.getString("condition");
mPatient = new Patient(patient_id, patient_name
           , patient_dni, patient_age, patient_gender
           , patient_created, patient_email
           , patient_height, patient_weight, condition);
mPatients.add(mPatient);
              }
```

After to get the patients array, to show the results we have added to the interface of the application a *RecyclerView* control, that we have filled by its adapter with the name and the last visit date of the patients obtained:

```
mAdapter = new PatientAdapter(getApplicationContext(),mPatients);
mAdapter = getAdapter();
mRecyclerView.setAdapter(mAdapter);
```

In Figure 4.8, it is showed the result after to run the GET request.

Figure 4.8: Full patients list screen

5

Classification of muscle activity in the lower back muscles

This chapter is divided into two main sections. Initially, it is presented in Section 5.1 a description about the tests assessment in a professional football team. In Section 5.2, are exposed the results obtained in three subsections: a first statistical approach in the subsection 5.2.1; Then, it is presented a clustering analysis 5.2.2 to check the possible data clusters by a study based on their features; Last step is to study the different known classifiers 5.2.3 and to apply them to the previous clusters established.

5.1 Methods

Surface electromyography (EMG) has been used to study fatigue as it provides a means of detecting this phenomenon (117). It has been reported that the capability to generate force in low back muscles decreases after prolonged exercise and is associated with lower M-wave amplitude, a change in activation level as indicated by root mean square of the EMG signal, and modification of the twitch contractile properties that results in decreased torque (118). Failure in force production may occur at various sites along the pathway from the central nervous system through to contractile components, and reduced energy stores, the build up of metabolic by-products and reduced central drive are thought to be important causative factors during fatiguing exercise (119). The present study aimed to analyze the EMG activity of the major muscle of the low back during intermittent exercise with a view to establishing the effects on EMG activity of an intermittent exercise protocol designed to induce fatigue as it occurs in soccer.

Seventeen professional soccer players of third division ranging from 19 to 32 years old, were recruited to be evaluated by one external physical therapist using the low back endurance test (STEET) by mDurance. Before performing the evaluation, the volunteers were informed about the research aims, risks and benefits of participation. All subjects were tested during the 2016/2017 Spanish competitive soccer preseason. The procedures were executed once per week to minimize the fatigue effects on the performance of the volunteers. The tests were realized both before as after of the trainings until a maximal of three times, in order to get a major variety in the results for to treat of to narrow the muscle fatigue intervals of each player. In both cases, each player had to fill a questionnaire (7.1) every time that he realized the test. This questionnaire was created and approved by the physical therapists who evaluated to the players. Its main goal was to evaluate as the players are perceived themselves physically and psychologically and to check what is their shape status for to can improve their performance. The tests were explained to the subjects before performing the sessions, assuring the full understanding of their phases. The STEET was performed as detailed in Section 3.1, while those involving the use of mDurance were carried out as described in Section 3.4.

5.2 Results description

5.2.1 Statistical analysis

The dataset which it is going to be analyzed is the set both the final results obtained and stored in the Sorensen class explained in the previous section 4.1.3 as the results obtained through the questionnaires. This dataset represents how the endurance and muscle fatigue affect to the soccer players for several attributes. At first, it is going to be summarized and described these variables:

- *Age, height and weight* are quantitative continue variables and they corresponding with the basic and personal information of each player. They are expressed in years, cm and Kg.

- *Body mass index (BMI)* is quantitative continue variable, which is calculated by the height (cm) and the weight (Kg).
- *Date*. It corresponds with the date when each test was realized.
- *Player's position*. It is describes the position which each player takes on the field.
- *Player's condition*. It is a qualitative ordinal variable. This represents the health condition of each player in the exact moment of to realize the test. This can take the following values: "Ok", "Fatigue", "Discomfort", "Injury" depending if the player was in perfect status, if he had done any diary activity which could cause muscle fatigue, if the player suffered some low back pain during the test and if the player was injured in that date.
- *Is_Played*. This indicates if the player who was going to do the test, he had played or not played. This, this is a qualitative ordinal binary attribute.
- *Q1, Q2 and Q4* represent the different questions of binary answer contained on the questionnaire. They describe both physical as the psychology status of the players for a particular date. They are also qualitative ordinal binary attributes.
- *Q3* is a quantitative discrete variable and it describes the hours number that each player was sited in that date. It is into the qualitative ordinal binary attributes.
- *From Q5 to Q10* are quantitative ordinal variables. They described the physic and mental shape status of each player in that date. Their corresponding levels for the questions are: Non fatigued, little fatigued, normal fatigued, very fatigued and fatigue discomfort.
- *Q11* is a quantitative ordinal variable. It describes the physic and mental endurance status of each player in that date. The unique difference with the previous is the levels for the questions: Non endurance, little endurance, normal endurance, good endurance and excellent endurance.

- *Likert scale.* It is the most widely used approach to scaling responses in survey research. When responding to a Likert questionnaire item, respondents specify their level of agreement or disagreement on a symmetric agree-disagree scale for a series of statements (120). Thus, the range captures the intensity of their feelings for a given item. A scale can be created as the simple sum of questionnaire responses over the full range of the scale. In so doing, Likert scaling assumes distances between each item are equal. Thus, the Likert's levels created with their corresponding ranking values from Q5 to Q10 questions are: Non fatigued (5), little fatigued (4), normal fatigued (3), very fatigued (2) and fatigue discomfort (1). For Q11 are Non endurance (1), little endurance (2), normal endurance (3), good endurance (4) and excellent endurance (5). After the questionnaire is completed, each item may be analyzed separately or may be summed to create a score for a group of items. Hence, Likert scales are often called summative scales. Thus, in this case the higher is the value obtained in the scale, a higher level of optimism respect to the endurance and the level of muscle fatigue will present a certain player subjectively. The maximal value for this scale in this case is 35.

- *Total time.* It is a quantitative discrete variable. It indicates the endurance total time in second, which the player lasted in his test.

- *Ratio.* It references to the ratio value mentioned in the Table 3.1 and calculated by the total time.

- *Ratio_label.* By the previous attribute, it is decided to label the resulted value with two levels: "Up" and "Down": The values are classified according to if it is higher than 1 or lower than 1. In the first case, it minds that the player is above the average and he is below in the opposite case.

- *RMS, ARV and MVC.* They are quantitative continue variables. They described the absolute muscle fatigue values measured in the end of the test by the application. They are expressed in mV.

- *RMS_per_second, ARV_per_second.* Another way to can compare the muscle activity among the players is to create a new attributes, which allow check that players had a major muscle activity. Thus, these variables are calculated dividing the final RMS and ARV values per the total time obtained in the test. They are expressed in mV/s.

- *Endurance label.* It is a quantitative ordinal variable. It describes the three possible endurance status that are registered by mDurance application in the Table 3.1. They are: "bad shape", "good shape", "excellent shape" according to the time duration in the test.

- *Fatigue label.* It is a quantitative ordinal variable. One of the difficulties that this work has encountered is to get to categorize the results of muscle activity obtained by the application. We could not find any bibliography which allowed us to label the muscle activity values. However, what if we have could classify were the different values obtained in function of if this muscle activity value was the minimal, the medium or the maximal value by a player. With this, we achieved to group these results in three levels: "minimal fatigue", "medium fatigue" and "maximal fatigue".

- *Fatigue per second label.* Finally, it is also decided to create a new EMG range based to categorize the results obtained by the RMS and ARV per second, in order to can classify to every players in function of the same unit time. The values are classified again in three levels: "minimum", "medium" and "maximal" fatigue per second.

It is also interesting to get some general information about the dataset. On the one hand, the dataset has a total of 44 instances and 25 attributes. In addition, a basic statistical analysis about these data is presented in the Table 5.1

In the data, it can be observed that it is a young team, since the 75% of the players have less of 26 years old. In addition, the table shows that the most of players are found into the healthy range with respect to the BCM. The bibliography indicates that a healthy man must be from 0.20 to 0.25 and the 75% of the players are found there.

Respect to the questions with binary answer (Q1, Q2 and Q4), these statistical indicate that in the most of cases, the players did not have any low back pain, they did not practice any activity which could cause him fatigue or discomfort and they felt themselves rested in the test date. It also can be observed that in Q3

Table 5.1: Statistics attributes

Statistic	Mean	St. Dev.	Min	Pctl(25)	Median	Pctl(75)	Max
Age	24.05	3.32	19	22	23	26	32
Height	178.07	5.21	164	176	178	183	185
Weight	74.39	5.19	62	72	75	79	81
BCM	0.23	0.01	0.22	0.23	0.23	0.24	0.27
Is_played	1.43	0.50	1	1	1	2	2
Q1	1.20	0.41	1	1	1	1	2
Q2	1.39	0.49	1	1	1	2	2
Q3	5.55	2.10	2	4	6	8	9
Q4	1.59	0.50	1	1	2	2	2
Q5	2.68	0.96	1	2	3	3	4
Q6	3.39	0.75	2	3	3	4	5
Q7	3.39	0.72	2	3	3	4	5
Q8	3.57	1.07	1	3	3.5	4	5
Q9	3.11	0.81	1	3	3	4	5
Q10	3.07	0.87	1	3	3	4	5
Q11	3.75	0.58	3	3	4	4	5
LikertScale	22.95	3.62	15	21	23	25.2	32
TotalTime	105.77	32.84	41	83.2	103.5	125.8	189
Ratio	1.10	0.34	0.42	0.86	1.07	1.30	1.96
RMS	0.17	0.09	0.05	0.13	0.16	0.18	0.64
ARV	0.13	0.07	0.04	0.10	0.12	0.14	0.50
MVC	0.67	0.36	0.23	0.52	0.62	0.68	2.01
RMS_sec	0.002	0.001	0.0003	0.001	0.002	0.002	0.01
ARV_sec	0.001	0.001	0.0002	0.001	0.001	0.001	0.01

is reflected that the most of players are sited a large number of hour (5.55 hours per day, becoming the 9 hours). With Q5, Q6 and Q7 are treated the physical load in some zones of the lower body. It is noteworthy that the muscle which received more load were the hamstring according to the noted by the players (2.68 in mean versus 3.39 and 3.57). In general, in the rest of questions (Q8-Q10) the players show high optimism level, since in all the cases, the mean is above of 3.0.

To highlight that the players are felt more optimism respect to endurance low back than with the fatigue low back, such as it appears in the results of the Q8 and 10. In general, the Likert scale affirms that more than 75% of the players are more optimism with their physical and mental shape in general.

If it is compared the values obtained from the first quartile to the maximal value with the Table 3.1, it is determined that at least, the 75% of tests realized are executed within the good and excellent range. However, the ratio attribute shows that almost the 50% of the players are below of the average that a healthy man lasts running the test.

It should be pointed for the RMS, ARV and MVC values the existing difference among the first and third quartiles with respect to the third and maximal value. This maximal value can affect to the final result in the subsequent results, so it must be treated with care. The same occurs with the RMS and ARV per seconds values.

The next step is visualized some graphics to check the visual data description:

An important attribute to highlight is the RMS, so it is going to be plotted with respect to the endurance and fatigue labels. The first figure (a) in Figure 5.1 shows that RMS values are greater when the players endurance is worse. This exposes a remarkable conclusion, it indicates that many of the players who realized the tests and that lasted a short time, they have obtained a higher RMS than the players who lasted much longer value. Thus, this is a clear indicator of existence of muscle fatigue in these players. In addition, it can be observed that between the "good" and the "excellent" time classification, the most of tests which had an "excellent" time classification obtained the lower RMS values. This supports the previous conclusion. Otherwise, the second figure (b) compares as the RMS per second values are distributed for the three fatigue classifiers. It can be observed that the results have much sense because show that the maximal, mean and minimal values of the "Maximal" cluster are bigger than these same

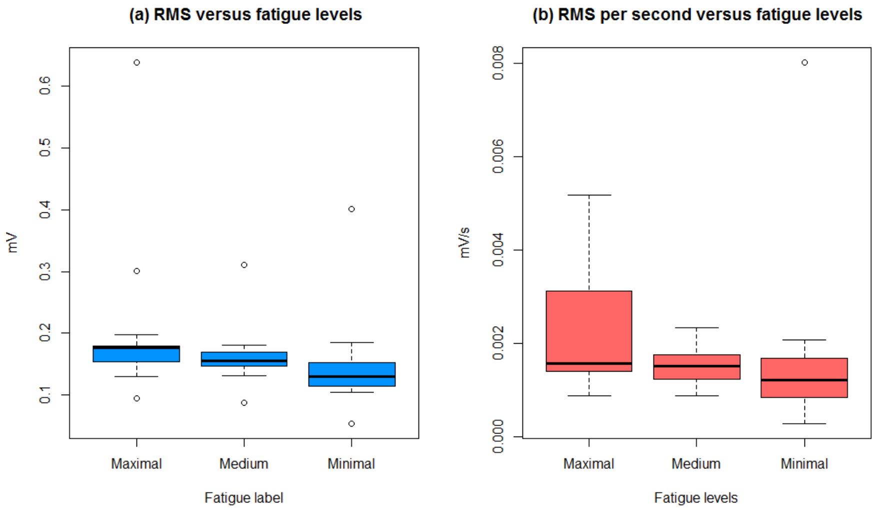


Figure 5.1: This figure describes as the RMS per second values are distributed respect to (a) endurance labels and (b) fatigue per second labels.

values in the "Medium" and "Minimal" cluster. In addition, these last are bigger than these in the "Minimal" cluster. However, the "Medium" cluster takes values both of the "Minimum" cluster as the "Maximal" cluster. This may difficult the supervised classification task.

In the next Figure 5.2 is represented the total time of duration of each test versus the Likert scale results (optimism values of the shape status) for each combination of endurance and fatigue labels in each quadrant. In addition, the players answer to the question 4 (are you rested today?). The results are also promising. It is showed as the most players who obtained an "Excellent and good shape" classification and that are in the "minimal" range of activity muscle, they answered that were rested in that date. Conversely, the most players who answered "NO" to the question are placed in the bottom and left quadrant, which is the "maximal" range of muscle fatigue measured by the application. In addition, the figure shows that the most of results obtained by the players who had an optimal perception about their physical and mental status (greater status in the Likert scale) are better classified than who indicated a the worst optimism answered. Another critical attribute, which it has been considered to classify the shape status of the players, it is if the player played or no played when he run the test (Is_played). Thus, by the Figure 5.3 is showed this classification. It can

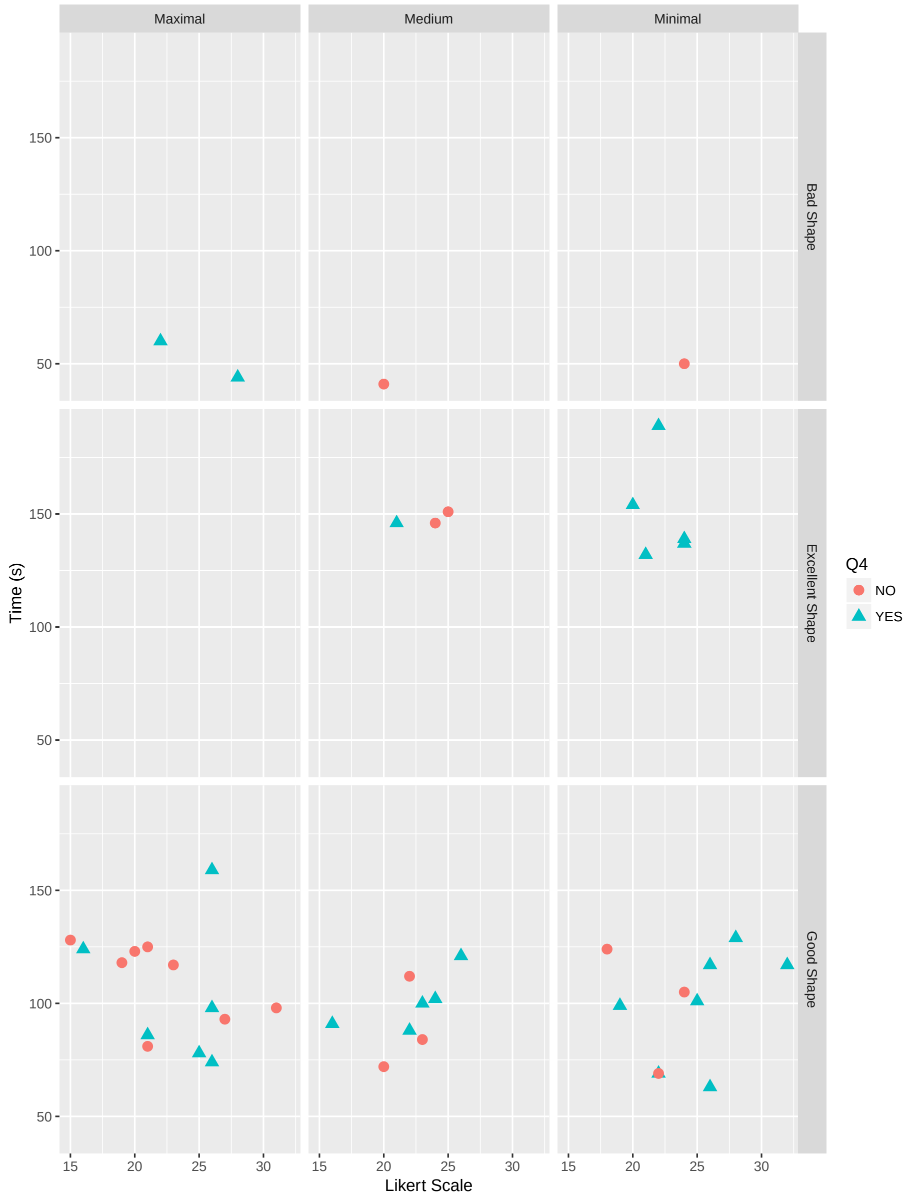


Figure 5.2: Scatter-plot of total time versus the Likert scale for each combination of endurance and fatigue labels in each quadrant. In addition, the answer to the question 4 is represented by shape and color.

be checked that the most players who have the best muscle activity classification had not played that day and thus they were not subjected to an intense physical activity. In addition, these players answered with more optimal answers than the

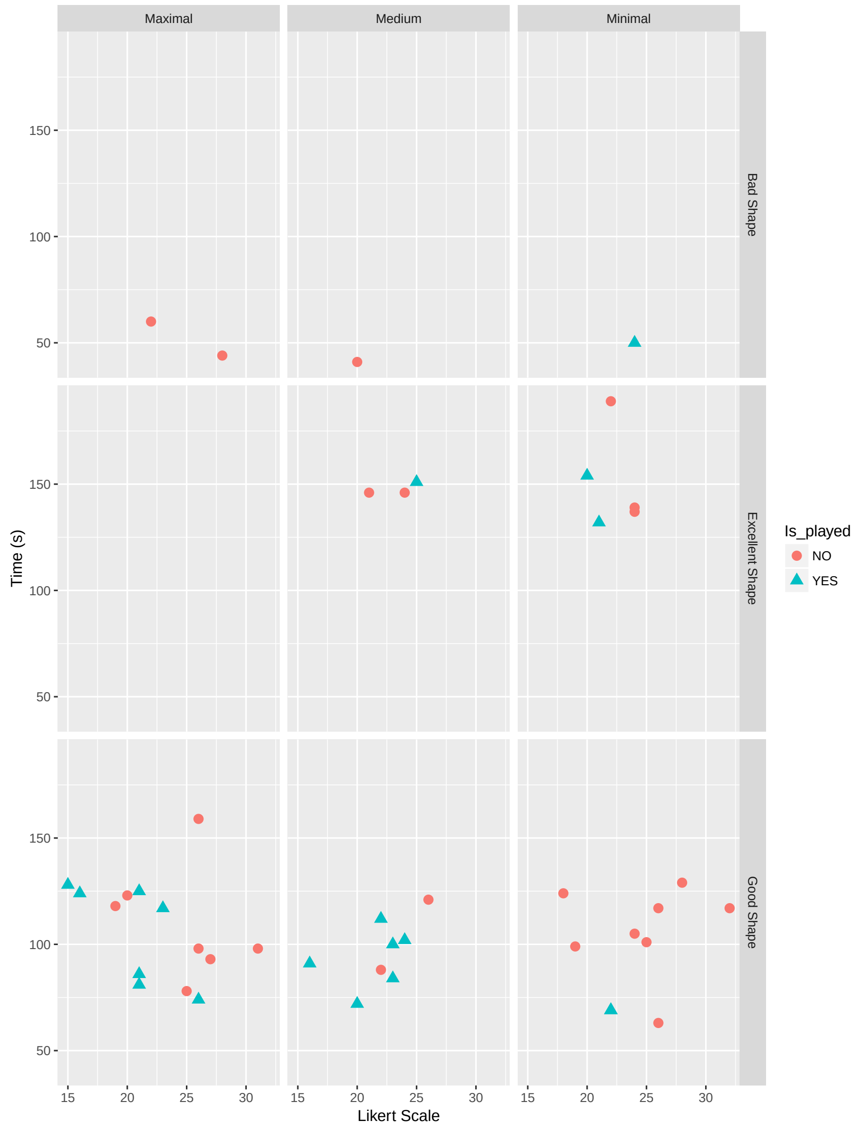


Figure 5.3: Scatter-plot of total time versus the Likert scale for each combination of endurance and fatigue labels in each quadrant. In addition, it is indicated if the played or not played by shape and color.

players who had trained that day. It is showed in the bottom and left quadrant. It also draws attention that the three of four tests classified as "bad shape" are of players who had not trained. It is usually due to that day, the player could have an injury or discomfort.

5.2.2 Clustering analysis

How it has been introduced in the previous section, both the endurance as the muscle activity would can be classified of two different ways:

- According the endurance. This classification is performed by the total time and the ratio obtained through the mDurance application. It knows that the endurance can be classified in bad, good and excellent shape according the total time and in UP or DOWN in function of the ratio. This way already was defined in (10) and mDurance does it automatically.

- According the muscle activity. It is a really difficult task because as it was mentioned in previous chapters, the muscle activity can be very different in each person. However, in the previous section, it aimed to give the first steps in this classification, defining: on the one hand, three muscle activity levels according if the player muscle activity measured (total RMS and ARV values in the end of the tests) was the minimal, medium or maximal that this player had obtained in one of the three tests that he had run; on the other hand, another three muscle activity levels weighted according to the total time obtained by the player in each test. Through this, they appears the minimal, medium or maximal RMS and ARV per seconds for each particular test.

According to these two ways, in this section is pretended to apply the known techniques of unsupervised classification methods (clustering) to extract the largest number of conclusion as it is possible.

In the clustering theory, the main step is the selection of the approximation functions. Of this choices depends the future results and the clustering quality. Moreover, it is important to select the attributes that are going to participate in the clustering process. It is had two very differentiable variables sets: the quantitative attributes refereed to the final results obtained for each tests and the qualitative attributes, which are refereed both the questionnaire results as the endurance and muscle activity descriptions. By that, it is must choose which variables are the most interesting. The feature extractions is essential by several reasons: it avoids to work with variables that do not give information for the learning task, the learning methods with less attributes will be more interpretable and some learning methods are very sensitive to the presence of variables significant and their performance may degrade significantly.

For this purpose, Chi-squared methods (121) and the correlation feature selection method (CFS) (122) were used. Finally, the selected attributes for to use in clustering were: RMS, RMS_sec representing to the quantitative variables and Condition, Q1, Q4, Ratio_label, Endurance_label and Fatigue_perSecond_label.

The next step is to decide what type of distance to choose. In general, it usually applies the Euclidean distance because is the most intuitive and offers excellent results when there compact groups. It must apply normalizing all the variables, since it ignores the different weights that could exist among them. Thus, it is had to normalize the data. However, for the qualitative attributes are going to be apply the binary distance (it must transform the Condition, endurance and muscle activity labels previously). Then, it is weighted the result.

The hierarchical clustering approach consists in to perform making nested successions of group in which the groups created in a level are fully contained in the following groups. Most algorithms are the agglomerative type, i.e, partitions start with as many groups as items and each step are united together. This approach could give a global idea about how many groups it is must consider. It has selected the Ward method (123), because realizing different experiments, it is which best results offered. Thus, it had to calculate the distance matrix of the data, because these algorithms are based to make transformations about this matrix decreasing its size.

A dendogram (124) is a visual tool, which is obtained after to make an hierarchical clustering. It helps to determinate the number of groups which would can represent the best data schema according the way in which the groups are going to be nested. In the Figure 5.4 is showed this schema after to apply Ward in the data. The number of groups in which are wanted to group the data is determined by the horizontal pruning. In the dendogram can be observe that if it is pruned for the height number four, it is had two cluster, from the height number three it would have three clusters and four cluster for the height number two. To select the best group,it is necessary to calculate the silhouette coefficient of the possible partitions. The silhouette coefficient (125) is a measure that combines the cohesion and separation. It compares the distance means among instances of a same group and the distance means among instances of different groups. The

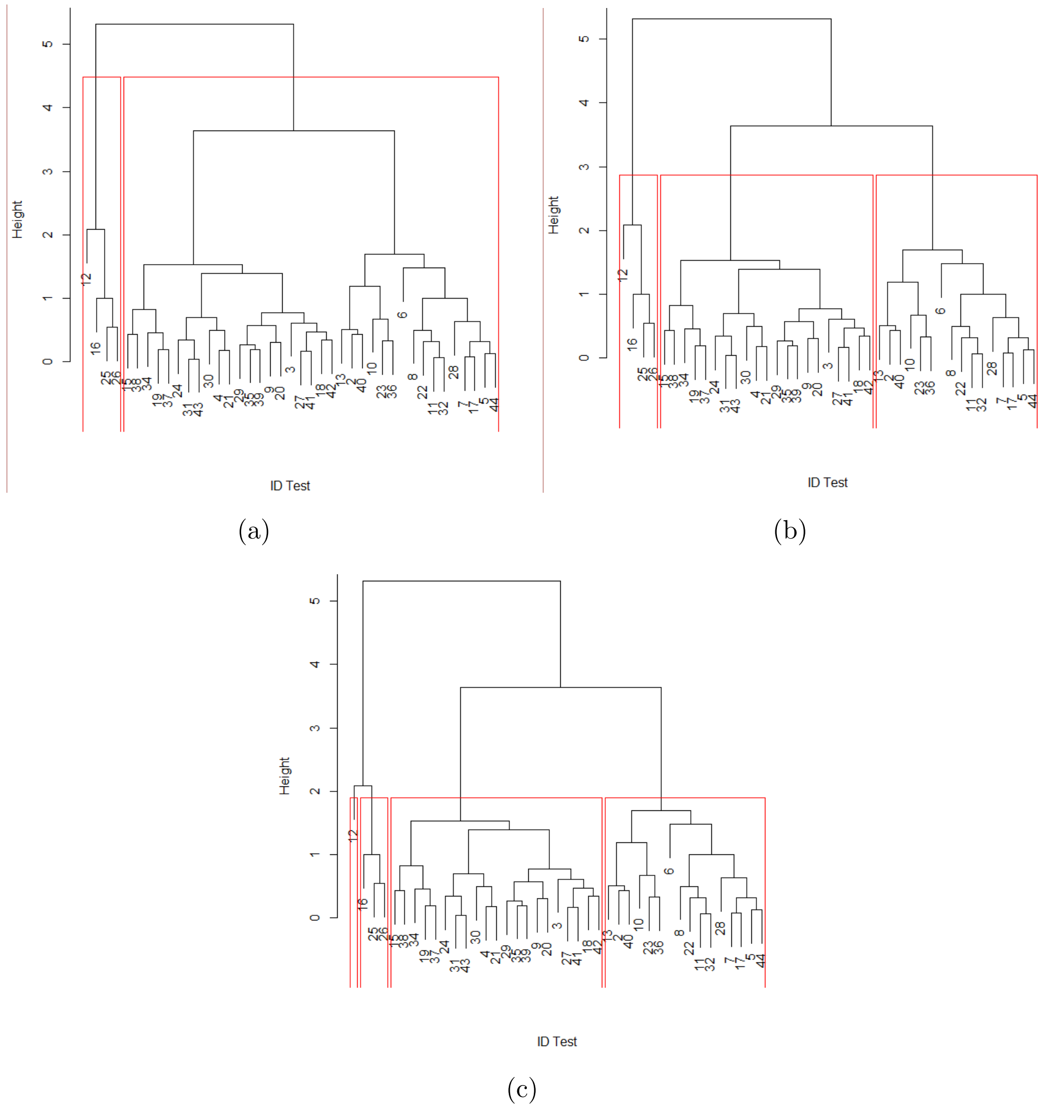


Figure 5.4: Ward analysis between the different clusters through Dendogram plots for: (**a**) two groups; (**b**) three groups; (**c**) four groups.

best scenario occurs when the distance among instances of the same group is small (cohesion) and the distance among instances of different groups are large (separation). The silhouette coefficient takes values between -1 and 1. Next to 1 are best the clusters. The results obtained are showed in Table 5.2 Analyzing the coefficients for the four groups, when it is taken only two groups (K=2), the cohesion and the separation are largest respect to three and four groups. However,

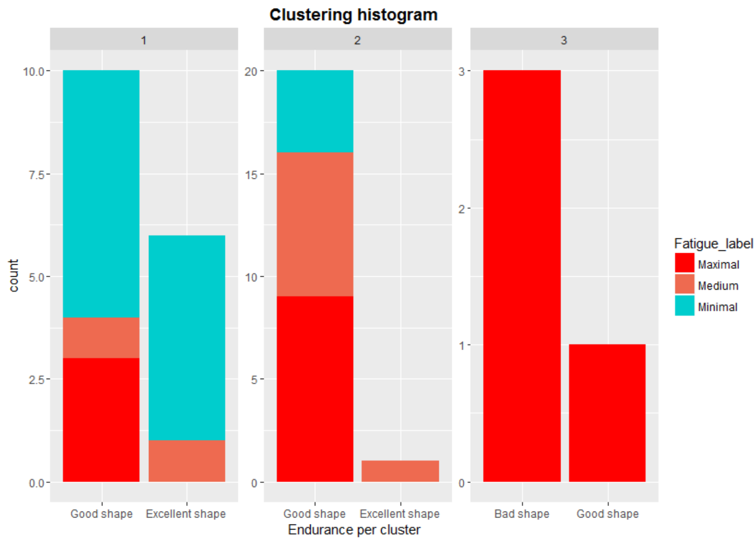
Table 5.2: Units per cluster and Silhouette coefficient for all the possible clusters in all the possible pruning (k=2, 3 and 4).

	Size	Ind. Silhouette width	Mean Silhouette width
K = 2			
Cluster 1	37	0.6022	0.5825
Cluster 2	4	0.3999	
K = 3			
Cluster 1	16	0.2648	0.3418
Cluster 2	21	0.4036	
Cluster 3	4	0.3251	
K = 4			
Cluster 1	16	0.2648	0.3456
Cluster 2	21	0.4023	
Cluster 3	1	0.0000	
Cluster 4	3	0.4932	

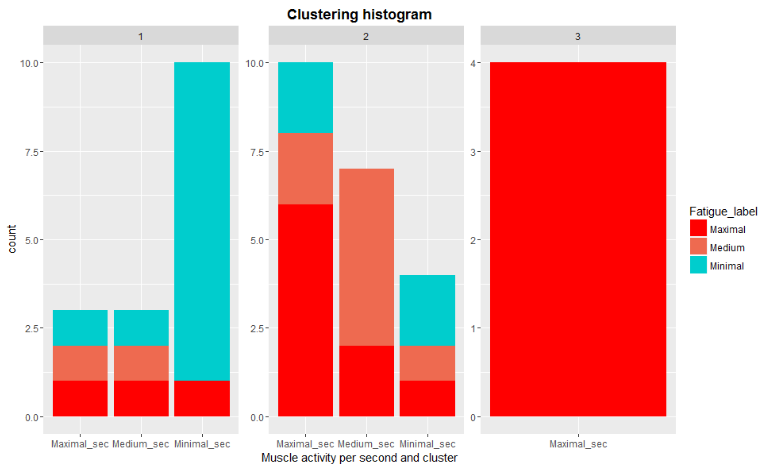
this happens by the little units that has the second cluster with respect to the first cluster. Thus, this cluster is not a good option. If they are compared the groups with three and four clusters respectively, it is observed that the Silhouette coefficients are very similar (0.3418, 0.3456). In the previous figure, 5.4(b) and 5.4(c) could see that both groups differ in one instance uniquely. It is not worth to take four cluster when only one clusters is for one unique instance. Thus, it is decided to take the three cluster options.

In addition, if the instances that belong to each clusters are checked in the dataset by its ID , it can be seen that all the instances belong to the cluster 3 are labeled as "Maximal" muscle activity, the most of the instances in the cluster 2 are labeled as "Maximal and Medium" muscle activity and the most of the instances belong to the cluster 1 are labeled into the "Medium and Minimal" muscle fatigue. This affirmation can be refuted by the Figure 5.5(b).

The players of the second cluster that have a major RMS value per second will have a major probability of to be labeled in "Maximal" or "Medium" muscle activity. Concretely, players who get a maximal RMS value per second will have a 81.25% of probability for to belong to the second or third cluster. Players who



(a)



(b)

Figure 5.5: Representation of the three cluster obtained by Ward methods through the histograms according to: (**a**) Endurance and muscle activity labels; (**b**) Muscle activity per second and global muscle activity labels.

obtained a medium RMS value per second will have a 77.77% for to belong to the second cluster versus to the first cluster. Finally, the probability for to belong to the first cluster when a players run a test with the minimal RMS value per second is of 73.33% and 26.77% for the second cluster.

Another aspect to note is the clustering of these instances according to the endurance labels and fatigue labels too. In Figure 5.5(a) is also demonstrated the accuracy of the classification used by mDurance for to label the RMS value in function of the player endurance. It is seen that the worst results obtained in the tests are corresponding with the instances of the third cluster, labeled as "Maximal muscle activity" instances. This aspects is extended to the second cluster, since it is observed as the endurance is growing progressively until the first cluster, where the most of the instances are grouped as good and excellent shape and "Minimal muscle activity". Another important aspect of this study is to check how it affects the opinion of the players with respect to their physical shape to the tests results. Although there were more attributes, for to make these clusters were used the "Q1, Q4, and IsPlayed" attributes due to its high correlation in test variables selection. To see how these variables have been classified into new groups, Figure 5.6 is used. For the first attribute shown in Figure 5.6(a) (Q1), it is detected that the representing instances when players claimed to have low back pain are distributed in the second and third clusters for "maximal and medium" muscle activity. This confirms the reliability of the proposed classification for this attribute. With respect to the responses of the players to Q4 5.6(b), there is homogeneity when they were distributed. However, it can be checked as in the first cluster is where the highest number of test classified with minimal fatigue were made by players who considered to be well rested. Finally to know how it affects have performed intense physical activity before performing the test, in Figure 5.7 is shown as instances are distributed according if the player had trained that day. Except the third cluster in which all the instances classified with maximum fatigue belong to players who had not trained. In the other two cluster, the labels are distributed from high to low fatigue for players who had trained that day. It should be noted this difference, since mDurance shows that for most instances could be classified correctly if a player had performed intense physical activity before each test.

5.2.3 Classification supervised analysis

Once it has been able to determine the number of groups that could belong a new RMS value obtained in the execution of a new test, it is time to try different classifiers referred to in the Section 2.3 and to check that accurate are. The supervised classification is one that allows classify a new instance in a certain known groups that previously have been labeled. To train different classification algorithms, it has been used the methodology of the R caret package (126). By *train* function, this package enables to optimize the various parameters that characterize a classifier. Figure 5.8 shows a diagram of the procedure, in which different configurations for the parameters are tested and chooses to get better results in a validation process. Cross-validation is a model validation technique for assessing how the results of a statistical analysis will generalize to an independent data set. Leave-p-out cross-validation (LpO-CV) involves using p observations as the validation set and the remaining observations as the training set. This is repeated on all ways to cut the original sample on a validation set of p observations and a training set. However, in this work is used Leave-one-out cross-validation (LOOCV) because there are not a large number of samples. This is a particular case of leave-p-out cross-validation with $p = 1$. Classification algorithms used can be grouped into two large families attending the goodness of the results:

- Individual classifiers. It is learning a single model, which can be of various kinds: KNN, SVM Linear, SVM Radial, SVM Polynomial, etc
- Assemblages classifiers. They are algorithms that add several previous classifiers, decision trees normally.

In the Table 5.3 are compared these groups on equal terms. It is taken like reference in each family the algorithms which have obtained better results among tested. Initially the same variables used for clustering were used. However, performing tests were checked that using only the RMS and RMS per second values, endurance and activity second labels, best accuracy was obtained. The statistical measures used to evaluate the accuracy and the quality of the classifiers in this work are:

- *Accuracy* (127) measures how correct is the label test identifies and excludes a given condition. Accuracy can be determined from sensitivity and specificity with the presence of prevalence.
- *Mean balanced accuracy* (127) measures the mean of all the accuracies obtained for each fold in LOOCV.

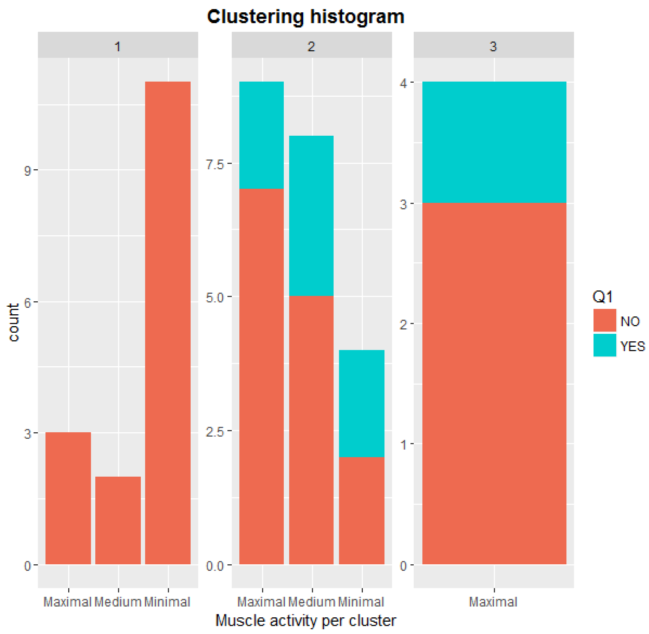
Table 5.3: Comparison between the main classifiers mentioned above.

Method	**Accuracy**	**Mean bal. accuracy**	**Mean Sensit**	**Mean Specif**
KNN (k = 9)	0.671	0.699	0.652	0.826
SVM Linear (C = 0.25)	0.658	0.733	0.640	0.825
SVM Radial (C=1,S=0.05)	**0.683**	**0.756**	**0.674**	**0.838**
CTree (minC = 0.01)	0.610	0.694	0.586	0.801

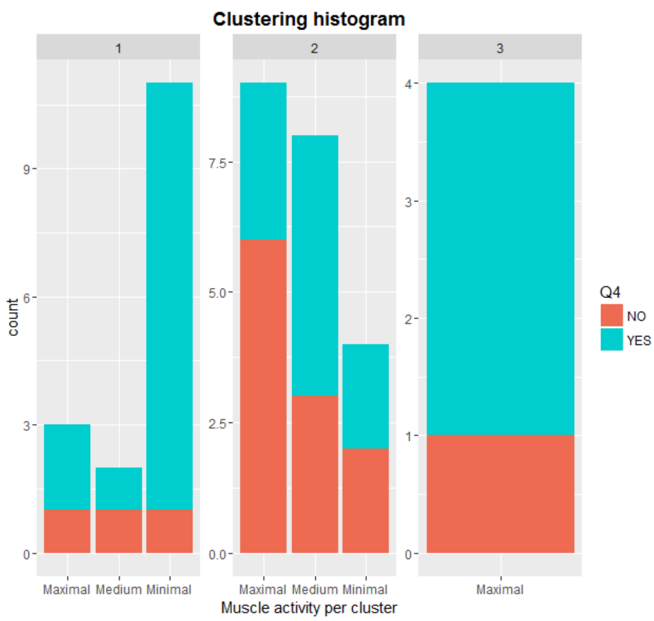
- *Mean sensitivity* (128) (also called the true positive rate) measures the proportion of positives that are correctly identified as such (for this case, the percentage of fatigued people who are correctly identified as having the condition).

- *Mean specificity* (128) (also called the true negative rate) measures the proportion of negatives that are correctly identified as such (for this case, the percentage of non-fatigued people who are correctly identified as not having the condition). It is noteworthy that both the sensitivity as the specificity are measured respect to the sensitivity and specificity means of each fold in LOOCV.

If they are compared all the statistical values, It may be concluded that the individual classifiers clearly outweigh the ensembles classifiers, and SVM Radial (with cost = 1 and sigma = 0.05) seems to be the most appropriate within these.



(a)



(b)

Figure 5.6: Distribution of the qualitative attributes through the histograms plots according to: (**a**) Question 1; (**b**) Question 2 in the three cluster obtained by Ward methods.

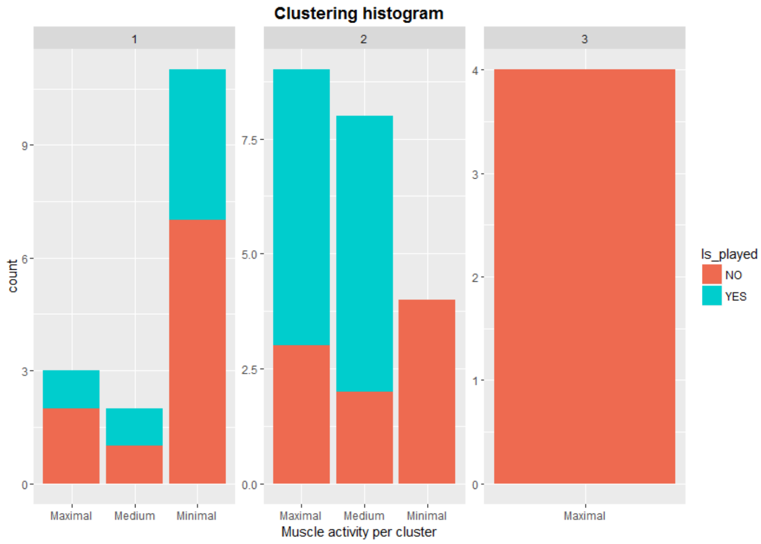


Figure 5.7: Distribution of isPlayed attribute in the three cluster obtained by Ward methods through the histograms plots.

```
Define sets of model parameter values to evaluate
for each parameter set do
    for each resampling iteration do
        Hold–out specific samples
        [Optional] Pre–process the data
        Fit the model on the remainder
        Predict the hold–out samples
    end
    Calculate the average performance across hold–out predictions
end
Determine the optimal parameter set
Fit the final model to all the training data using the optimal parameter set
```

Figure 5.8: Pseudocode representing the caret integrated package to optimize the parameters of the algorithms classification procedure.

6

Conclusions

Low back pain remains a major cause of labor dissertation in the world. Specifically, Spain is the second. In addition, each passing year, the symptoms start at a younger age, so the forecasts are becoming more unfavorable time. Moreover, the proliferation of new wearable and portable electromyography systems allow the monitoring of the muscle activity is increasingly common; and the inclusion for years leading smartphones to simplify the task of electromiography systems adapt to any area: health, fitness, professional sports, injury prevention, etc. However, electromyography applied to the characterization of muscle activity remains largely unknown due to the difficulty of to classify the values obtained in fatigued or non-fatigued. According to this problem, the most experts in the field have begun to use data mining techniques to characterize and support those results.

However, most of these have just used it to determine which techniques are most significant to sort the muscle activity. It is with that, why in this work has been presented mDurance as a tool that also to help assess the endurance tests, can store a record both temporary results related to endurance and electromyographic related to the activity of the muscles trunk through a new back-end service that has been incorporated into the system. In addition, since the endurance tests are over time, it is wanted to implement a new windowing temporary sliding system to facilitate the work of the specialist to monitor the behavior of the muscles during the test and provide greater feedback to the user. All data is sent to the server through an API REST that speeds up the storing of all the data, including personal information users and specialists. Since the information handled is confidential, it has been included a new authentication mechanism Token based ensures secure communication for any requests made to the server.

Finally, in order to use the knowledge acquired during the master of science data, it has conducted a study about the behavior of the muscles of the lumbar area on a professional football team. In addition, taking advantage of the new system back-end implemented as data mining engine, they have produced the first groupings of what could be a new unsupervised classification system of the lumbar muscle activity. Its preparation was also made using a questionnaire with which was intended to obtain feedback on their own physical condition of the players and study how their perceptions approaches reality. For this section, it must be concluded that the results claim that clusters correspond to the values of low-medium and medium-maxim muscle activity, which initially started in the study activity, further demonstrating that muscle activity shown by the players was in most cases consistent with their perception. Once it was possible to group instances into three groups, they were applied the most famous supervised classification techniques, both individual and ensembles, in order to test the accuracy of the classification of new instances. It must be concluded that the highest accuracy, specificity and sensibility were obtained with a single classifier, such as SVM Radial, which obtained a 68.3%, 67.4% and 83.8% in precision and sensitivity and specificity respectively.

6.1 Future work

Given the important magnitude of the fields treated in this work and due to the good reception that mDurance has gotten during this time, they are proposed both new improvements for the system and new future studies lines to continue with this work:

- *System functionalities*. In this field, we are going to include some ideas to develop:

  - Guidelines with personalized exercises to realize according to the patient endurance trunk or its muscle fatigue. It could provides a new application functionality to reproduce YouTube videos as well as local stored ones.

- A specialist dashboard. Taking advantage of the back-end service, it could be implement a dashboard based front-end technologies that speeds up both work to treat data as its visualization, allowing the specialist to obtain other feedback regarding the analysis of the data stored in the system using this platform. It is proposed the following novel library to develop it: Plotly (129), which is also based Python and it would facilitate its development.

- New data mining libraries implemented in Android. In the future could be possible to get the same classification proposed in this work through of an Android smartphone. In addition, this would allow to the specialist get new conclusions and feedback about the health muscle status of the patients. Some libraries observed for this purpose are: AndroidLibSVM (130), KNN (131) or decision trees by Weka (132).

- It is important to adapt new systems to the "Big data era". We believe that it is important to adapt the system to new non-relational data bases (NoSQL), necessary to implement Big Data techniques in the future. Some of the databases that might interest us are: Apache Cassandra (133) or MongoDB (134). Currently existing libraries for Android which implement algorithms Big Data in the databases mentioned, for example Spark (135).

- To incorporate new temporal windows in the system. The literature contains a wide variety of time windows applied to electromyographic signals processing. Featured are: multiple Hamming windows (MHW), multiple trapezoidal windows (MTRW) and multiple Slepian windows (MSW) (136, 137).

- Implementation of mDurance in another devices. It would be interesting that this application could also run and automatically adjust to the tablets, with both the results and graphs, as texts and help videos would be displayed better. Moreover a tablet could be a more suitable device for this application thinking that it will be used by a specialist.

• *Features extractions.* In this work, it was used a new approach based questionnaires. However, this may be an ineffective tool if the respondents are not objective. In addition, if this happens it could incorporate noise to classify the following endurance tests correctly. Therefore, the following methods are proposed to improve the robustness of the system:

- Incorporate tracking devices to people who realize the tests, for example, bracelets or smart watches that allow to generate greater reliability of the physical condition who have previously performed these people: sleep, physical activity, if they are rested, etc.

- Add new devices that allow us to obtain other variables to improve the classification of new tests, for example infrared cameras that get the temperature of the muscles during the test.

- New endurance tests. There are another endurance test to assess both trunk muscles and another kind of muscles of the human body. For example, the one leg standing balance test or the hip extension (87) associated to the past, present and future LBP or the muscle imbalance. The active straight leg raising test (87) associated to the postpartum sacroiliac pain.

- New endurance and muscle fatigue features. Other way is to include new endurance and muscle fatigue parameters in mDurance, such as the mean frequency (MNF) and median frequency (MDF) of EMG signal. In this way, the specialist would have more accurate data to diagnose the patient. Another important constant in LPB chronic patients is the phenomenon of flexion/extension during the complete flexion of the trunk. Thus, it is interesting to introduce the different relations between flexion and extension in the final assessment from each test. It can do it if we compare the times among STEET and TCSET or among both sides in SBET. The extensor/flexor ratio is a more sensitive index than peak torque in predicting future back pain in asymptomatic individuals (87).

- *Application fields.* We propose some field where mDurance is interesting to be used:

  - Investigation fields. They exist a several possible investigation ways to develop in the future based mDurance. For example, it may be assess the mDurance working in low back pain chronic patients. To observe the endurance and muscle fatigue that they show during the test and to compare with healthy subjects.

  - Fitness area. Due to continued growth in the fitness field , it could be interesting o incorporate new sensors to the system (ECG, EEK, etc) and to implement a new method is able to monitor the proper activity of people who play sports without any assistance.

- Another interested area could be study the behavior of this system in animals. For example, if we also think in sport area, we can link them in the study of the endurance and the muscle fatigue in race horses, dogs, etc.

7

Appendix

In this appendix, we will show the questionnaire used to evaluate the perception of the physical condition of each player about himself.

The questionnaire includes the following personal information and questions about the players: the player's name, the date of assessment of each test and if the player had trained or played before or not.

Then, it consists of three questions binary responses (Q1, Q2 and Q4) of possible discomfort or fatigue lumbar or limiting physical activities that may affect the player's performance during the test. A numerical question about the number of hours that the player had been sitting on that date (Q3) and seven questions about their muscle perception into some lower back and lower body muscles.

Nombre ______________________________

Fecha ______________________________

¿Ha jugado? ______________________________

ID		SI	NO
Q1	¿Tienes hoy molestias lumbares?		
Q2	¿Has realizado hoy alguna actividad que te haya podido causar fatiga en la espalda?		
Q3	¿Cuántas horas estimas que has pasado hoy sentado? (Indicar horas)		
Q4	¿Te sientes hoy descansado?		

ID		NADA	POCA	NORMAL	MUCHA	MOLESTIAS
Q5	¿Cómo valoras la carga muscular acumulada en tus isquios?					
Q6	¿Cómo valoras la carga muscular acumulada en tus abductores?					
Q7	¿Cómo valoras la carga muscular acumulada en tus cuádriceps?					
Q8	¿Cómo valoras tu fatiga lumbar de hoy cuando te has despertado?					
Q9	¿Cómo valoras tu fatiga lumbar en este momento?					
Q10	¿Cómo valoras tu fatiga en general?					
Q11	¿Cómo valoras hoy tu resistencia lumbar?					EXTRAORDIONARIA

Figure 7.1: Questionnaire used for the physical and mental player assessment before and after of each endurance test.

References

[1] IHME. *Institute for Health Metrics and Evaluation (IHME).* June 2015. xv, 2, 3, 4, 5, 6

[2] IHS Technology. *IHS Technology.* June 2015. xv, 10, 11, 12

[3] IHS Technology. **IHS Wearable contributing to improved clinical outcomes and better work**. Available online: `https://www.ihs.com/pdf/`, June 2016. xv, 11

[4] Hande Türker and Hasan Sözen. **Surface electromyography in sports and exercise**. 2013. xv, 16

[5] K. Muller and H. Welsch. **BioZen**. Available online: `http://www.kurzweilai.net/biofeedback-mobile-app/`, 2010. Accessed: 2016-03-30. xv, 17

[6] Athos. **Live Athos**. Available online: `https://www.liveathos.com/`, 2016. Accessed: 2016-03-30. xv, 20

[7] Myonthec. **Myontec, intelligent clothing**. Available online: `https://http://www.myontec.com/`, 2016. Accessed: 2016-03-30. xv, 20, 22

[8] Wooyoung Kim. **Parallel clustering algorithms: survey**. *Parallel Algorithms, Spring*, 2009. xv, 23

[9] Rafael de Castro Aguiar. **Myoelectric Prosthesis-Modelization and Control of a Bionic Arm**. 2015. xv, 33

[10] Ignacio Diaz Reyes. *mDurance, an innovative method to accurately assess the trunk endurance tests.* PhD thesis, University of Granada, Granada, 2015. 1, 35, 36, 38, 44, 47, 67, 95

[11] Oresti Banos, Jose Antonio Moral-Munoz, Ignacio Diaz-Reyes, Manuel Arroyo-Morales, Miguel Damas, Enrique Herrera-Viedma, Choong Seon Hong, Sungyong Lee, Hector Pomares, Ignacio Rojas, et al. **mDurance: A Novel Mobile Health System to Support Trunk Endurance Assessment**. *Sensors*, **15**(6):13159–13183, 2015. 1, 36, 38

[12] Laxmaiah Manchikanti, Vijay Singh, Frank JE Falco, Ramsin M Benyamin, and Joshua A Hirsch. **Epidemiology of low back pain in adults**. *Neuromodulation: Technology at the Neural Interface*, **17**(S2):3–10, 2014. 3

[13] Mugdha Gore, Alesia Sadosky, Brett R Stacey, Kei-Sing Tai, and Douglas Leslie. **The burden of chronic low back pain: clinical comorbidities, treatment patterns, and health care costs in usual care settings**. *Spine*, **37**(11):E668–E677, 2012. 4

[14] Bernabé Esteban, Álvaro Tejeda-Lorente, Carlos Porcel, José Antonio Moral-Munoz, and Enrique Herrera-Viedma. **Aiding in the Treatment of Low Back Pain by a Fuzzy Linguistic Web System**. In *Rough Sets and Current Trends in Computing*, pages 250–261. Springer, 2014. 4, 9

[15] Wei Fan and Albert Bifet. **Mining big data: current status, and forecast to the future**. *ACM sIGKDD Explorations Newsletter*, **14**(2):1–5, 2013. 5

[16] Kathy Kincade. **Data mining: digging for healthcare gold**. *Insurance & Technology*, **23**(2):2–7, 1998. 5

[17] T Christy. **Analytical tools help health firms fight fraud**. *Insurance & Technology*, **22**(3):22–26, 1997. 6

[18] Steve Biafore. **Predictive solutions bring more power to decision makers.** *Health Management Technology*, **20**(10):12, 1999. 7

[19] Michael Silver, Taiki Sakata, Hua-Ching Su, Charles Herman, Steven B Dolins, Michael J O Shea, et al. **Case study: how to apply data mining techniques in a healthcare data warehouse**. *Journal of healthcare information management*, **15**(2):155–164, 2001. 7

[20] Andrea Benko and Ben Wilson. **Online decision support gives plans an edge**. *Managed Healthcare Executive*, **13**(5):20, 2003. 7

[21] G Gillespie. **There's gold in them thar'databases.** *Health data management*, **8**(11):40–4, 2000. 7

[22] HR Kolar. **Caring for healthcare.** *Health management technology*, **22**(4):46–47, 2001. 7

[23] Anne Milley. **Healthcare and Data Mining Using data for clinical, customer service and financial results**. *Health Management Technology*, **21**(8):44–45, 2000. 7

[24] Hian Chye Koh, Gerald Tan, et al. **Data mining applications in healthcare**. *Journal of healthcare information management*, **19**(2):65, 2011. 8, 9

[25] Neelamadhab Padhy, Dr Mishra, Rasmita Panigrahi, et al. **The survey of data mining applications and feature scope**. *arXiv preprint arXiv:1211.5723*, 2012. 8

[26] Lynn Fieltz and David Scott. **Prediction of physical performance using data mining**. *Research Quarterly for Exercise and Sport*, **74**(1):1–25, 2003. 8

[27] Osama K Solieman. **Data mining in sports: A research overview**. *Dept. of Management Information Systems*, 2006. 9

[28] José A Moral-Muñoz, Bernabé Esteban-Moreno, Manuel Arroyo-Morales, Manuel J Cobo, and Enrique Herrera-Viedma. **Agreement between face-to-face and free software video analysis for assessing hamstring flexibility in adolescents.** *The Journal of Strength & Conditioning Research*, 2015. 9

[29] Fuensanta Palacín-Marín, Bernabé Esteban-Moreno, Nicolas Olea, Enrique Herrera-Viedma, and Manuel Arroyo-Morales. **Agreement between telerehabilitation and face-to-face clinical outcome assessments for low back pain in primary care**. *Spine*, **38**(11):947–952, 2013. 9

[30] Tamara Hafner and Jeremy Shiffman. **The emergence of global attention to health systems strengthening**. *Health policy and planning*, **28**(1):41–50, 2013. 9

[31] Marcia M Ditmyer, Robert Topp, and Matthew Pifer. **Prehabilitation in preparation for orthopaedic surgery**. *Orthopaedic Nursing*, **21**(5):43–54, 2002. 9

[32] Emily Oshima Lee and Ezekiel J Emanuel. **Shared decision making to improve care and reduce costs**. *New England Journal of Medicine*, **368**(1):6–8, 2013. 10

[33] Trevor G Russell, Peter Buttrum, Richard Wootton, and Gwendolen A Jull. **Rehabilitation after total knee replacement via low-bandwidth telemedicine: the patient and therapist experience**. *Journal of telemedicine and Telecare*, **10**(suppl 1):85–87, 2004. 10

[34] Santosh Kumar, Wendy Nilsen, Misha Pavel, and Mani Srivastava. **Mobile health: Revolutionizing healthcare through transdisciplinary research**. *Computer*, (1):28–35, 2013. 10

[35] Adam C Powell, Adam B Landman, and David W Bates. **In search of a few good apps**. *JAMA*, **311**(18):1851–1852, 2014. 10

[36] GL Kidd and JA Oldham. **An electrotherapy based on the natural sequence of motor unit action potentials: a laboratory trial**. *Clinical Rehabilitation*, **2**(2):125–138, 1988. 15

[37] Gary Kamen and David Gabriel. *Essentials of electromyography.* Human kinetics, 2010. 16

[38] Robert Richer, Peter Blank, Dominik Schuldhaus, and Bjoern M Eskofier. **Real-Time ECG and EMG Analysis for Biking Using Android-Based Mobile Devices**. In *2014 11th International Conference on Wearable and Implantable Body Sensor Networks*, pages 104–108. IEEE, 2014. 18

[39] Michael Sung and Alex Pentland. **Livenet: Health and lifestyle networking through distributed mobile devices**. *Proceedings WAMES*, 2004. 18

[40] Shyamal Patel, Chiara Mancinelli, Anthony Dalton, Ben Patritti, Trudy Pang, Steven Schachter, and Paolo Bonato. **Detecting epileptic seizures using wearable sensors**. In *2009 IEEE 35th Annual Northeast Bioengineering Conference*, pages 1–2. IEEE, 2009. 18

[41] Nahla F Abdel-Maboud, Bassant Mohamed ELBagoury, Mohamed Roushdy, and Abdel-Badeeh M Salem. **EMGNEU: Mobile health application for neuromuscular disosrders diagnosis**, 2012. 18

[42] Alexandros Pantelopoulos and Nikolaos G Bourbakis. **A survey on wearable sensor-based systems for health monitoring and prognosis**. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, **40**(1):1–12, 2010. 19

[43] Shimmer. **Shimmer EMG**. Available online: `http://www.shimmersensing.com/shop/shimmer3-emg-unit`, 2016. Accessed: 2016-03-30. 19

[44] Deltason. **Biometrics Data Acquisition Systems**. Available online: `http://www.deltason.com/products`, 2015. Accessed: 2015-03-30. 20

[45] Myo. **Myo armband**. Available online: `https://www.myo.com/`, 2016. Accessed: 2016-03-30. 21

[46] Sparkfun. **MyoWare Muscle Sensor**. Available online: `https://www.sparkfun.com/products/13723`, 2016. Accessed: 2016-03-30. 21

[47] Vidameter. **Vidameter, a smartband for your health**. Available online: `http://www.vidameter.com/`, 2016. Accessed: 2016-03-30. 21

[48] Vylyv. **Vylyv, Kegel exercise shorts for men**. Available online: `http://www.vylyv.com/`, 2016. Accessed: 2016-03-30. 21

[49] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. **Data clustering: a review**. *ACM computing surveys (CSUR)*, **31**(3):264–323, 1999. 22, 23, 24

[50] Anil K Jain and Richard C Dubes. *Algorithms for clustering data.* Prentice-Hall, Inc., 1988. 23, 24

[51] Peter HA Sneath, Robert R Sokal, et al. *Numerical taxonomy. The principles and practice of numerical classification.* 1973. 23

[52] Benjamin King. **Step-wise clustering procedures**. *Journal of the American Statistical Association*, **62**(317):86–101, 1967. 23

[53] Ricardo A Baeza-Yates. **Introduction to Data Structures and Algorithms Related to Information Retrieval.**, 1992. 24

[54] Richard C Dubes. **How many clusters are best?-an experiment**. *Pattern Recognition*, **20**(6):645–663, 1987. 24

[55] James MacQueen et al. **Some methods for classification and analysis of multivariate observations**. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, **1**, pages 281–297. Oakland, CA, USA., 1967. 24

[56] Arthur P Dempster, Nan M Laird, and Donald B Rubin. **Maximum likelihood from incomplete data via the EM algorithm**. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977. 24

[57] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. **A density-based algorithm for discovering clusters in large spatial databases with noise.** In *Kdd*, **96**, pages 226–231, 1996. 25

[58] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. *Automatic subspace clustering of high dimensional data for data mining applications*, **27**. ACM, 1998. 25

[59] Lotfi A Zadeh. **Fuzzy sets**. *Information and control*, **8**(3):338–353, 1965. 25

[60] Joseph C Dunn. **A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters**. 1973. 25

[61] James C Bezdek. *Pattern recognition with fuzzy objective function algorithms.* Springer Science & Business Media, 2013. 25

[62] Sotiris B Kotsiantis, I Zaharakis, and P Pintelas. **Supervised machine learning: A review of classification techniques**, 2007. 26, 27

[63] J Ross Quinlan. *C4. 5: programs for machine learning.* Elsevier, 2014. 26

[64] Johannes Fürnkranz. **Separate-and-conquer rule learning**. *Artificial Intelligence Review*, **13**(1):3–54, 1999. 26

[65] Aijun An and Nick Cercone. **Rule quality measures improve the accuracy of rule induction: an experimental approach**. In *International Symposium on Methodologies for Intelligent Systems*, pages 119–129. Springer, 2000. 27

[66] William W Cohen. **Fast effective rule induction**. In *Proceedings of the twelfth international conference on machine learning*, pages 115–123, 1995. 27

[67] Celia C Bojarczuk, Heitor S Lopes, and Alex A Freitas. **Discovering comprehensible classification rules using genetic programming: a case study in a medical domain**. In *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation-Volume 2*, pages 953–958. Morgan Kaufmann Publishers Inc., 1999. 27

[68] Aijun An and Nick Cercone. **Discretization of continuous attributes for learning classification rules**. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 509–514. Springer, 1999. 27

[69] Guoqiang Peter Zhang. **Neural networks for classification: a survey**. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, **30**(4):451–462, 2000. 27

[70] Robert J Howlett and Lakhmi C Jain. *Radial basis function networks 2: new advances in design*, **67**. Physica, 2013. 28

[71] Finn V Jensen. *An introduction to Bayesian networks*, **210**. UCL press London, 1996. 29

[72] Tom M Mitchell. **Machine learning. 1997**. *Burr Ridge, IL: McGraw Hill*, **45**:37, 1997. 29

[73] Christopher JC Burges. **A tutorial on support vector machines for pattern recognition**. *Data mining and knowledge discovery*, **2**(2):121–167, 1998. 30

[74] Konstantinos Veropoulos, Colin Campbell, Nello Cristianini, et al. **Controlling the sensitivity of support vector machines**. In *Proceedings of the international joint conference on AI*, pages 55–60, 1999. 30

[75] Bernhard Schölkopf and Christopher JC Burges. *Advances in kernel methods: support vector learning*. MIT press, 1999. 30

[76] Marc G Genton. **Classes of kernels for machine learning: a statistics perspective**. *Journal of machine learning research*, **2**(Dec):299–312, 2001. 30

[77] Thomas G Dietterich. **An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization**. *Machine learning*, **40**(2):139–157, 2000. 30

[78] Leo Breiman. **Bagging predictors**. *Machine learning*, **24**(2):123–140, 1996. 31

[79] Yoav Freund and Robert E Schapire. **A desicion-theoretic generalization of on-line learning and an application to boosting**. In *European conference on computational learning theory*, pages 23–37. Springer, 1995. 31

[80] Eric Bauer and Ron Kohavi. **An empirical comparison of voting classification algorithms: Bagging, boosting, and variants**. *Machine learning*, **36**(1-2):105–139, 1999. 31

[81] J Ross Quinlan. **Boosting first-order learning**. In *International Workshop on Algorithmic Learning Theory*, pages 143–155. Springer, 1996. 31

[82] Tianqi Chen and Carlos Guestrin. **Xgboost: A scalable tree boosting system**. *arXiv preprint arXiv:1603.02754*, 2016. 31

[83] Paolo Bonato, Delsey M Sherrill, David G Standaert, Sara S Salles, and Metin Akay. **Data mining techniques to detect motor fluctuations in Parkinson's disease**. In *Engineering in Medicine and Biology Society, 2004. IEMBS'04. 26th Annual International Conference of the IEEE*, **2**, pages 4766–4769. IEEE, 2004. 32

[84] Ana Aguilera, Alberto Subero, and Ramón Mata-Toledo. **Application of data mining techniques on EMG registers of hemiplegic patients**. In *Industrial Conference on Data Mining*, pages 254–265. Springer, 2013. 32

[85] Abdulhamit Subasi. **Classification of EMG signals using PSO optimized SVM for diagnosis of neuromuscular disorders**. *Computers in biology and medicine*, **43**(5):576–586, 2013. 33

[86] Krzysztof Brzostowski and Maciej Zieba. **Analysis of human arm motions recognition algorithms for system to visualize virtual arm**. In *Systems Engineering (ICSEng), 2011 21st International Conference on*, pages 422–426. IEEE, 2011. 34

[87] Craig Liebenson. **Spinal stabilization–an update. Part 2functional assessment**. *Journal of Bodywork and Movement Therapies*, **8**(3):199–210, 2004. 36, 108

[88] Craig Liebenson. **Spinal stabilization-an update. Part 1-biomechanics**. *Journal of Bodywork and Movement Therapies*, **8**(2):80–84, 2004. 36

[89] F. Biering-Sorensen. **Physical measurements as risk indicators for low-back trouble over a one-year period.** *Spine*, **9**(2):106–119, 1984. 36

[90] Mary T Moffroid. **Endurance of trunk muscles in persons with chronic low back pain: assessment, performance, training**. *Journal of Rehabilitation Research and Development*, **34**:440–447, 1997. 36

[91] Kerrie Evans, Kathryn M Refshauge, and Roger Adams. **Trunk muscle endurance tests: reliability, and gender differences in athletes**. *Journal of Science and Medicine in Sport*, **10**(6):447–455, 2007. 36

[92] Michael P Reiman, Amber D Krier, Julie A Nelson, Michael A Rogers, Zachariah O Stuke, and Barbara S Smith. **Comparison of different trunk endurance testing methods in college-aged individuals**. *International Journal of Sports Physical Therapy*, **7**(5):533, 2012. 36

[93] Christophe Demoulin, Marc Vanderthommen, Christophe Duysens, and Jean-Michel Crielaard. **Spinal muscle evaluation using the Sorensen test: a critical appraisal of the literature**. *Joint Bone Spine*, **73**(1):43–50, 2006. 36

[94] Wolfgang Gruther, Franziska Wick, Birgit Paul, Christoph Leitner, Martin Posch, Michael Matzner, Richard Crevenna, and Gerold Ebenbichler. **Diagnostic accuracy and reliability of muscle strength and endurance measurements in patients with chronic low back pain**. *Journal of Rehabilitation Medicine*, **41**(8):613–619, 2009. 36

[95] GyuTae Kim, Mohammad A Ahad, Mohammed Ferdjallah, and Gerald F Harris. **Correlation of muscle fatigue indices between intramuscular and surface EMG signals**. In *SoutheastCon, 2007. Proceedings. IEEE*, pages 378–382. IEEE, 2007. 37, 38, 39

[96] Juliana M Fernández, Rubén C Acevedo, and Carolina B Tabernig. **Influencia de la fatiga muscular en la señal electromiográfica de músculos estimulados eléctricamente**. *Revista EIA*, (7):111–119, 2007. 38

[97] Abdullah Al Zaman, T Sharmin, Mohammad Ashraf Ali Khan, and M Ferdjallah. **Muscle fatigue analysis in young adults at different MVC levels using EMG metrics**. In *SoutheastCon, 2007. Proceedings. IEEE*, pages 390–394. IEEE, 2007. 38, 39

[98] Sijiang Du and Marko Vuskovic. **Temporal vs. spectral approach to feature extraction from prehensile EMG signals**. In *Information Reuse and Integration, 2004. IRI 2004. Proceedings of the 2004 IEEE International Conference on*, pages 344–350. IEEE, 2004. 39

[99] Dawn T MacIsaac, Philip A Parker, Kevin B Englehart, and Daniel R Rogers. **Fatigue estimation with a multivariable myoelectric mapping function**. *IEEE Transactions on Biomedical Engineering*, **53**(4):694–700, 2006. 39

[100] Guilherme Macedo and Christina Thorpe. **Performance Study of Cloud Computing Back-End Solutions for Mobile Applications**. In *2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC)*, pages 327–332. IEEE, 2015. 40

[101] Heroku. **Inc Heroku cloud application platform**. Available online: `http://www.heroku.com`, 2016. Accessed: 2016-03-30. 40

[102] Openshift. **Inc. red hat openshift online marketplace**. Available online: `https://marketplace.openshift.com/home`, 2016. Accessed: 2016-03-30. 40

[103] Amazon. **Amazon aws elastic beanstalk - application management - paas**. Available online: `https://aws.amazon.com/es/elasticbeanstalk/`, 2016. Accessed: 2016-03-30. 40

[104] Jason H Christensen. **Using RESTful web-services and cloud computing to create next generation mobile applications**. In *Proceedings of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, pages 627–634. ACM, 2009. 40, 41

[105] Roy Thomas Fielding. *Architectural styles and the design of network-based software architectures*. PhD thesis, University of California, Irvine, 2000. 40

[106] Jeffrey C Mogul, Jim Gettys, Henrik Frystyk, Tim Berners-Lee, and Roy T Fielding. **Hypertext Transfer Protocol–HTTP/1.1**. 1997. 41

[107] Django. **Django documentation**. Available online: `https://docs.djangoproject.com/en/1.10/`, 2016. Accessed: 2016-03-30. 41

[108] NGIX. **Ngix documentation**. Available online: https://nginx.org/, 2016. Accessed: 2016-03-30. 42

[109] WSGI. **WSGI documentation**. Available online: https://wsgi.readthedocs.io/en/latest/, 2016. Accessed: 2016-03-30. 42

[110] Gunicorn. **Gunicorn documentation**. Available online: http://gunicorn.org/, 2016. Accessed: 2016-03-30. 42

[111] Django. **Django Rest Framework**. Available online: http://www.django-rest-framework.org/, 2016. Accessed: 2016-03-30. 42

[112] PostgreSQL. **PostgreSQL Database**. Available online: https://www.postgresql.org/, 2016. Accessed: 2016-03-30. 43

[113] MariaDB. **MariaDB Database**. Available online: https://mariadb.org/download/, 2016. Accessed: 2016-03-30. 43

[114] Tatsuo Ishii. **Guide to Complete Mastering PostgreSQL**. *Gijutsu–Hyohron Co., Japan*, 2001. 43

[115] Oresti Banos, Rafael Garcia, Juan A Holgado-Terriza, Miguel Damas, Hector Pomares, Ignacio Rojas, Alejandro Saez, and Claudia Villalonga. **mHealthDroid: a novel framework for agile development of mobile health applications**. In *International Workshop on Ambient Assisted Living*, pages 91–98. Springer, 2014. 47

[116] **Sqlite**. Available online: https://sqlite.org/, 2016. Accessed: 2016-03-30. 47

[117] B Bigland-Ritchie, R Johansson, and JJ Woods. **Does a reduction in motor drive necessarily result in force loss during fatigue**. *Biochemistry of Exercise (International Series on Sport Sciences Vol. 13)*, pages 864–870, 1983. 85

[118] Anne Michaut, MICHEL Pousson, NICOLAS Babault, and Jacques Van Hoecke. **Is eccentric exercise-induced torque decrease contraction type dependent?** *Medicine and science in sports and exercise*, **34**(6):1003–1008, 2002. 85

[119] H Westerblad, S Duty, and DG Allen. **Intracellular calcium concentration during low-frequency fatigue in isolated single fibers of mouse skeletal muscle**. *Journal of Applied Physiology*, **75**(1):382–388, 1993. 85

[120] Gail M Sullivan and Anthony R Artino Jr. **Analyzing and interpreting data from Likert-type scales**. *Journal of graduate medical education*, **5**(4):541–542, 2013. 88

[121] James E Higgins and Gary G Koch. **Variable selection and generalized chi-square analysis of categorical data applied to a large cross-sectional occupational health survey**. *International Statistical Review/Revue Internationale de Statistique*, pages 51–38, 1977. 96

[122] Mark A Hall. **Correlation-based feature selection of discrete and numeric class machine learning**. 2000. 96

[123] Vladimir Batagelj. **Generalized Ward and related clustering problems**. *Classification and related methods of data analysis*, pages 67–74, 1988. 96

[124] Evelyn C Pielou. *The interpretation of ecological data: a primer on classification and ordination.* John Wiley & Sons, 1984. 96

[125] Peter J Rousseeuw. **Silhouettes: a graphical aid to the interpretation and validation of cluster analysis**. *Journal of computational and applied mathematics*, **20**:53–65, 1987. 96

[126] Max Kuhn. **The caret package**. Available online: http://topepo.github.io/caret/index.html, 2016. Accessed: 2016-03-30. 101

[127] Wen Zhu, Nancy Zeng, Ning Wang, et al. **Sensitivity, specificity, accuracy, associated confidence interval and ROC analysis with practical SAS® implementations**. *NESUG proceedings: health care and life sciences, Baltimore, Maryland*, pages 1–9, 2010. 101

[128] Tom Fearn. **Sensitivity and specificity**. *NIR news*, **20**(4):16, 2009. 102

[129] Plotly. **Plotly library**. Available online: https://plot.ly/python/, 2016. Accessed: 2016-03-30. 107

[130] Yctung. **AndroidLibSVM**. Available online: https://github.com/yctung/AndroidLibSvm, 2016. Accessed: 2016-03-30. 107

[131] Sergio. **KNN-android**. Available online: https://github.com/sergioivan154/KNN-android, 2016. Accessed: 2016-03-30. 107

[132] Rjmarsan. **WEKA for Android.** Available online: https://github.com/rjmarsan/Weka-for-Android, 2016. Accessed: 2016-03-30. 107

[133] APACHE. **Apache Cassandra**. Available online: `http://cassandra.apache.org/`, 2016. Accessed: 2016-03-30. 107

[134] MONGODB. **MongoDB official site.** Available online: `https://www.mongodb.com/`, 2016. Accessed: 2016-03-30. 107

[135] APACHE. **Apache Spark, Github repository**. Available online: `https://github.com/spark/spark-sdk-android`, 2016. Accessed: 2016-03-30. 107

[136] ANGKOON PHINYOMARK, PORNCHAI PHUKPATTARANONT, AND CHUSAK LIMSAKUL. **Feature reduction and selection for EMG signal classification**. *Expert Systems with Applications*, **39**(8):7420–7431, 2012. 107

[137] G VENUGOPAL, M NAVANEETHAKRISHNA, AND S RAMAKRISHNAN. **Extraction and analysis of multiple time window features associated with muscle fatigue conditions using sEMG signals**. *Expert Systems with Applications*, **41**(6):2652–2659, 2014. 107