POLITECNICO DI TORINO

Electronic Engineering

AN ELECTRONIC SYSTEM TO ASSESS SLEEP DISORDERS

Author:
Dimroci Enea

Supervisors:
Olmo Gabriella
RECHICHI Irene

October 2020

Contents

AN ELECTRONIC SYSTEM TO ASSESS SLEEP DISORDERS

Abstract

Sleep disorders has shown to be one of the pre-clinical marker for the development of neurodegenerative disease. Between these disease, the Parkinsonism (Parkinson disease form) is the one much more connected with sleep disorders, around 70% of people that shows some kind of sleep disorder has proved to develop Parkinsonism within five years. The standard way to measure some kind of sleep disorder is the Polysomnography. This exam concerns on a multi parametric test and its goal is to discriminate among the sleep stages of the patient. The main problems of this solution are the invasivity of the machinery used to perform the exam, and the amount of hospital resources, which cannot accommodate the amount of people that needs to perform this exam. In order to meet the demand one solution might be to move on smart devices for the data acquisition, in this way the invasivess and the hospital resources problems can be overcame. The problem is that the parameters to track are a lot, and the goal is to reduce the invasivess first. A reduced amount of sensing elements can be used if some correlation among data is found. The analysis solution might be the AI analysis which has shown great results in all analysis problems. The new paradigm of training models with raw data acquired from sensors and labels given as input, or clustering analysis of input data, might show correlations among data that cannot be found by pathologists. An other problem is that, if the invasiveness is overcame, and people might acquire their sleep parameters directly from home, the amount of data to be analyzed increases, and it became necessary to develop an infrastructure to allow this scalability. The goal is to show the entire flow, focusing on the design of the data acquisition system and the firmware configuration of the smart devices to acquire raw data, the integration of data coming from different smart devices into standard structure to be processed, the analysis phase of these heterogeneous data with AI algorithms, and also the storage into databases of raw data and results coming from the analysis of them. All these phases are integrated to be driven by a GUI interface to help and to support pathologist during the diagnosis.

Chapter 1

General Introduction

1.1 Introduction

Sleep quality has shown to be a marker for different fields, it impacts a lot on people performances, and it may also have an impact on people health. Specifically sleep disorders has proved to be a constant in many cases of neurodegenerative disease, in particular for Parkinson disease. The gold standard for sleep measurement is the polysomnogram (PSG), which requires a sleep lab, sleep technician, and monitoring of multiple physiological parameters. As such, polysomnography is generally restricted to the assessment of sleep for only one or two nights. Longitudinal, ambulatory sleep measurement can benefit a number of populations, including patients with suspected sleep disorders, workers in occupations where any impairment in alertness is high risk, and healthy individuals who desire improved sleep for maximal cognitive and physical performance and optimal health.

The current method accepted by the medical and scientific community for objective, longitudinal sleep measurement in the ambulatory setting is actigraphy. Actigraphy refers to the use of FDA-approved (Food and Drug Administration), wrist worn accelerometry devices that measure movement to estimate sleep. A large body of peer-reviewed evidence has assessed performance of actigraphy against PSG. However, actigraphy has significant inadequacies that limit its use: actigraphs are expensive compared to the consumer sleep trackers which are already owned by millions of individuals, actigraphs record only movement, and they struggle to correctly classify wake events during the attempted sleep period.

Consumer marketed wearable devices are a tempting solution to the problem of ambulatory sleep tracking given ease of use, widespread availability, measurement of multiple biological signals, low cost, and opportunity for integration with other health technology products. However, the minimal validation of consumer sleep trackers and their associated outputs against PSG has precluded use in clinical, research, and occupational settings. Even when devices are validated against PSG once, both device firmware

and associated software are frequently updated by the manufacturer. As algorithms that determine sleep metrics are rarely disclosed, such updates could make previous validation studies irrelevant. These barriers to validation and the lack of transparency surrounding the associated software's sleep scoring methods have historically reduced enthusiasm for consumer marketed wearable use in medicine and research. Overcoming these barriers is of great interest, as a growing body of evidence has begun to reveal the potential clinical and research utility of commercially available products.
On top of the rapid progress in sensor development, technological advances have expanded our ability to analyze the vast amount of data they collect. Machine learning techniques and other advanced computational methods that make use of the current capabilities of computing power, memory, and storage to classify novel input data are well suited for the prediction of sleep metrics from massive amounts of sensor acquired signals. Therefore, the weighted sum algorithms that have formed the cornerstone of existing actigraphy software programs are likely to be out performed by newer techniques.

1.1.1 Parkinson's Disease

Parkinson's Disease (PD) is a progressive neurological condition. This means that it causes problems in the brain and gets worse over time and it affects a considerable number of people all over the world.
Motor and non-motor symptoms are the results of the dopaminergic neurons death, and this dopamine lack leads to the classical parkinsonian movements such as tremor, irregular gait, paralysis and a low muscular strength.
There are also other kinds of symptoms that cannot be directly related to the motor system. In particular brain activity and heart rate data during sleeping may show relevant features related to the patient and its sleep behavior.
In order to keep track of these multiple informations, a lot of devices that sense different parameters have to be employed during the sleep disorder analysis.
Obviously, this requires multisensing devices able to collect data during the sleep.
It also requires efficient algorithms that analyze these data coming from a variety of sensors, and give a result in terms of sleep activity so that physicians may monitor the disease progress.
Polysomnography is the standard exam to diagnose and monitor Sleep Disorders for monitoring sleep disorders.
This technology encompasses different sensors that keep track of the main parameters during sleeping such as the *movement index*, the *heart rate*, the *EEG* signals and other signals related to different features.
Physicians are able to classify the different sleep stages analyzing the data coming from all devices. The results are stored in the Analysis Software and form the patients

database. Upon request, raw data can be exported in order to carry out research studies. Through the analysis of medical reports and follow-up tests, the Physician or Sleep Neurologist may carry out a more precise diagnosis.
Sleep disturbances are common in Parkinson's disease and comprise the entire spectrum of sleep disorders. It leads also to the development of insomnia and daytime sleepiness. Sleep regulation relies the complex and integrated function of multiple brain areas and neurotransmitters, many of which have also been shown to be affected in patients with Parkinson disease (PD).
Besides the PD-related impairments of brain and neurotransmitter function there are other major contributing factors to disturbances of sleep and wakefulness in patients with PD. These include dopaminergic drugs, which are known to influence regulation of sleep and wakefulness, as well as other medications symptoms that impair patients' sleep, such as nocturnal akinesia, and genetic factors that predispose to disturbances of sleep and wakefulness. In patients with PD, both the sleep macrostructure and sleep microstructure, manifesting as disturbed integrity of certain sleep stages are affected, for instance the disturbed sleep spindles and K-complexes, or insufficient muscle atonia during REM sleep.
The categories of sleep disturbances affecting patients with PD thus comprise insomnia, disorders of daytime somno- lence, sleep-related breathing disorders, circadian disorders, and sleep-related movement disorders, namely restless legs syndrome (RLS), and parasomnias.
The activity during sleep is impaired, with subsequent manifestation of parasomnias (mainly REM sleep behavior disorders, but also, albeit more rarely, sleepwalking, and overlap parasomnia). Restlesslegs syndrome has been reported to be frequent in patients with Parkinson's disease, although there is no consensus on whether it is more frequent in Parkinson's disease than in the general population.
In order to monitor sleep parameters in PD patients, wearable, multisensor, consumer devices are now commonplace.
The idea is to design an infrastructure that allows PD patients to record sleep parameters directly from their own home. Without so much invasivity like with the state-of-the-art standards to monitor sleep disorders such as polysomnography.
Is also necessary to speed up the data analysis in order to keep track efficenty of patients parameters, and possibly to predict future stages related to PD patients, so in order to slow down as much as possible the incoming healthy problems.
Thinking about the large amount of people that can perform such measurements, is necessary to build an infrastructure that is able to continuously analyze incoming data from thousands of patients.
This system has to provide results from raw data analysis, and store these results in a database, that will be consulted by pathologists in order to make the right decision for different patients in a reduced amount of time.

1.2 State of Art

In order to perform *Sleep Disorder Analysis*, the state of art, and nowadays the de facto standard, is the *Polysomnography* (PSG) exam.
Polysomnography in a sleep laboratory is time-consuming and expensive. With the evolution of technology, portable devices have emerged that measure more or less the same sleep variables in sleep laboratories as in the home.
This exam is performed by trained staff and concerns the extrapolation of physiological data from patients using different sensors.
The PSG collects the following biosignals:

- Electroencephalogram (EEG) signal for the brainwaves evolution,
- Electrooculogram (EOG)signal for understanding eyes movements,
- Electromyogram (EMG) signal for measuring the skeleton muscle activity overnight,
- Electrocardiogram (ECG) for recording heart rate.
- Motor signals to track the movement index.
- Blood oxidation

From the integration of these biosignals, the pathologist reports a diagnosis for the patient with results in terms of relevant features for sleep disorder cases.
Some of these relevant features may be the total sleep time (TST), the number of awakenings during the night, the sleep onset time (SOT) which is the time duration from full awakeness to sleep, the REM sleep duration and many other features.

1.3 Problems

Polysomnography (PSG) is powerful in terms of quality and accuracy of the analysis, but there are many problems related to this method, for example the invasiveness of the equipment used to perform this exam is one of the main problems. This issue may lead to the first night effect, that corrupt in some way the analysis.
Another issue regards the resources available in the hospital to perform this exam, to overcome this problem, one solution is to employ smart devices that can substitute the big instrumentation needed in PSG.
Hence patients may perform their exams at home, without the need of performing the exam at a sleep laboratory.
These problems are further explained below:

- Invasiveness
  To collect all the data needed for the analysis, a lot of sensing devices become necessary.
  As previously described, the EEG signal for the brainwaves evolution is needed, also EOG and EMG signals for understanding eyes movements and recording heart rate respectively can help in understanding sleep disorders.
  Motion data are really important to track the body movement over the night.
  Therefore, all of these requirements lead to a lot of invasiveness on the patient during this exam:

Figure 1.1: Polysomnography Invasiveness

- First Night Problem
  In-laboratory polysomnography (PSG) is performed in a different environment than the patients' usual one. Therefore, it may result in worse sleep quality. This is called the first night effect (FNE) and is defined by longer sleep onset latency, lower sleep efficiency, longer REM latency, decreased REM, and increased alpha (Tamaki et al., 2005). On the other hand, some patients may sleep better than usual which is called the reverse first night effect (RFNE). The problem is that raw data might be corrupted by these effects.

- Limited Hospital Resources
  The resources made available by the hospital cannot accommodate many people, the analysis are spread over time. It's necessary to accelerate this phase and the idea is to move on smart devices to perform data acquisition during sleep.

Chapter 2

Flow to Alternative Solutions

2.1 Introduction

The aim of this project is to show which solutions may be adopted to overcome the problems related to the PSG exam.
I want also to show the entire flow: from acquiring data and storing it, to data analysis them with different techniques and providing results to pathologists in order to speed up the diagnosis and overcome the problems previously explained.
A first step is necessary to solve the invasiveness of this multiparametric sensing system because it leads to one main problem: the sleepness of the patient is corrupted by these devices attached to their body, obviously different from the they normally sleep, but this affects the recorded data.
To sum up, data have a bias that is not negligible. To overcome this problem , the idea is to use modern low cost sensors that may be packaged into a very small area, reducing the physical connections of the overall system to the patient, and thus the invasiveness.
Making this change, the goal is to reach the performance of the standard system in the PSG exam.
For this work, the main focus is the motion data, paving the way not only to the integration of other PSG elements, but also suggesting some ideas on how this can be done.This work also refers to one paper that integrates heart rate records in the study.
The work flow starts with the choice of the board used to track data.
The sensing elements needed to perform an *Inertial Analysis* of the patient are:

- Accelerometer
- Gyroscope
- Magnetometer

The board chosen is the *SensorTile*, a *'STMicroelectronics'* board that integrates all the features needed for this work, and also other elements for futher improvements that will be explained in detail in this thesis.
Chosen the device, the next step is the *Firmware Development* of the board. This part of the flow is fundamental to take data coherently with the study that is planned.
The *Embedded C* code is developed from an already existing firmware, provided by 'STMicroelectronics'; the parameters of the sensors are changed and different configurations on the processor and the RF modules are applied to achieve the wanted result in terms of Power Consumption and processed data.
The power consumption is really important in this case because of the duration of the exam, ranging from 6 hours to 8 hours of continous sampling and processing.
So this code is is used to go through the toolchain to write the machine code in the memory device.
STMicroelectronics provides an IDE *'STM32CubeIDE'* and a microcontroller *'STM-NUCLEO'* with which you can go through the toolchain, converting the C code in machine code that runs on the specific hardware with the specific instruction set of the board processor. After that, you can write the binaries in the memory device to run the application. Data can be monitored in real time with an *Android* application *'ST-BLE Sensor'*. Is possible to take data from Bluetooth or save data to an SD card. In order to take the data stored on the device or on the Android application, you can simply send data through different apps including Telegram or your e-mail in a really easy way.
At this point raw data are available in some way for the analysis.
To make this analysis different *Python* scripts are developed, the goals of these scripts are various and involve different steps from *Data Preprocessing* to *Models Definition*, from *Statistical Models* to *AI Models*, from *Clustering* to *Classification* and finally to the report of the results of this analysis.
Also other Python scripts are developed to easily see the results and the details about the analysis through a *GUI Interface* that make simple the navigation among different data and results.
Another feature implemented is the storage of raw data and results in a *SQL Database*, and this brings scalability on the flow because in this way you can store a very large amount of data, making this database also a source of raw data that can be used as dataset for future studies.

2.2 Hardware Description of the Board

2.2.1 Sensortile Board

The board chosen is the *SensorTile* from *STMicroelectronics*, it required characteristics in terms of sensing elements, processing element and storage resources.
It also has other sensors such as barometer and microphone, that can be integrated for further studies and analysis.
The integration of all these modules is done in a very small area: 13.5 mm x 13.5 mm. This characteristic impacts on the invasiveness of the whole structure used to perform the exam on patients. The most important modules needed to track physical quantities for the inertial analysis are integrated in a really small device.
The main components of this chip are shown in the following figure:

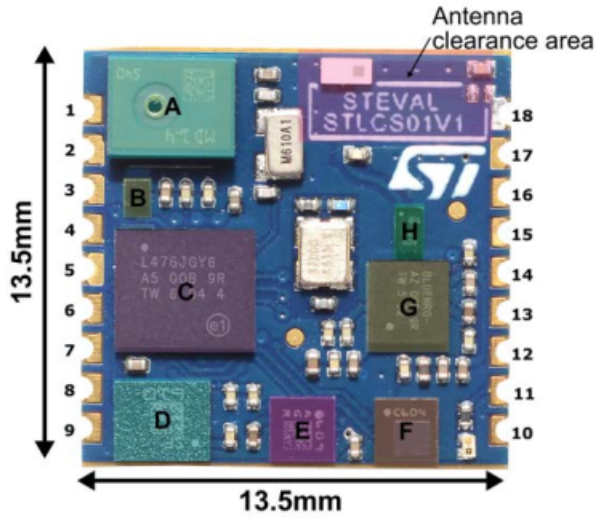


Table 1. **STEVAL-STLCS01V1 main components**

Reference	Device	Description
A	MP34DT05-A	MEMS audio sensor digital microphone
B	LD39115J18R	150 mA low quiescent current low noise LDO 1.8 V
C	STM32L476 MCU	ARM Cortex-M4 32-bit microcontroller
D	LSM6DSM	iNEMO inertial module: low-power 3D accelerometer and 3D gyroscope
E	LSM303AGR	Ultra-compact high-performance eCompass module: ultra-low power 3D accelerometer and 3D magnetometer
F	LPS22HB	MEMS nano pressure sensor: 260-1260 hPa absolute digital output barometer
G	BlueNRG-MS	Bluetooth low energy network processor
H	BALF-NRG-02D3	50 Ω balun with integrated harmonic filter

Figure 2.1: SensorTile Platform

A functional block diagram of the System On Chip (SoC) can be seen in the following figure:

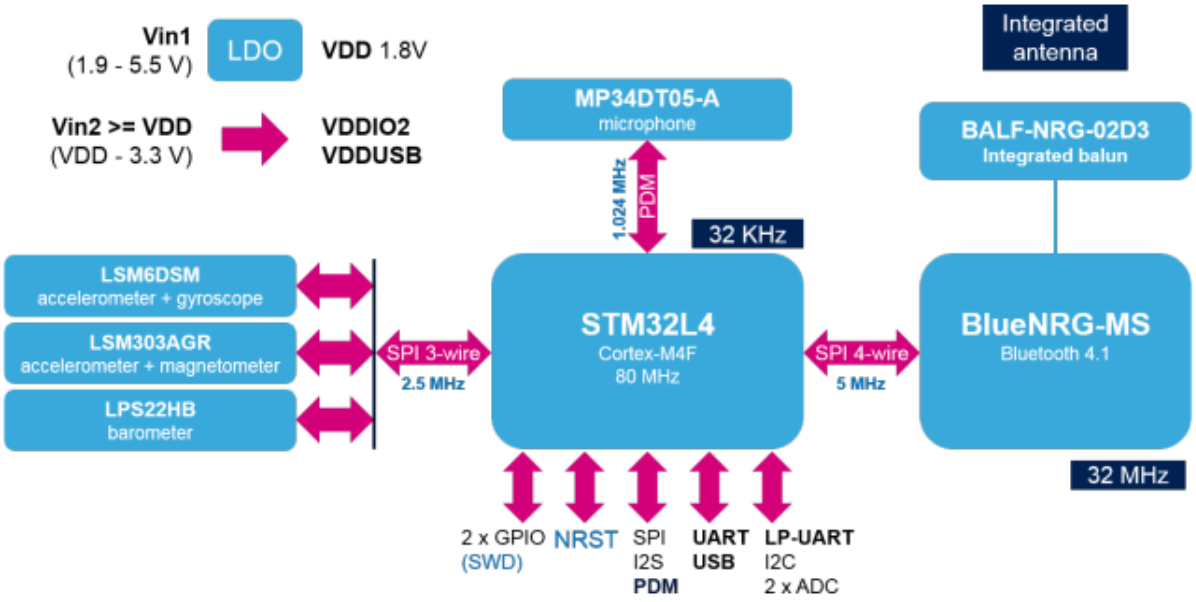


Figure 2.2: SensorTile Functional Block Diagram

This board is a highly integrated platform with a lot of functions aimed at improving the system design cycles and accelerating the delivery of results.

2.2.2 Overview of the Main Components

2.2.2.1 LDO Voltage Regulator

The LD39115J provides 150 mA maximum current from an input voltage ranging from 1.5 V to 5.5 V with a typical dropout voltage of 80 mV. It is stabilized with a ceramic capacitor. The ultra low drop voltage, low quiescent current and low noise features make it suitable for low power battery-powered applications. Power supply rejection is 65 dB at low frequencies and starts to roll off at 10 kHz. An enable logic control function puts the LD39115J in shutdown mode allowing a total current consumption lower than 1 µA. The device also includes a short-circuit constant current limiting and thermal protection.

2.2.2.2 STM32L476xx

The STM32L476xx devices are the ultra-low-power microcontrollers based on the high-performance Arm® Cortex®-M4 32-bit RISC core operating at a frequency of up to 80 MHz. The Cortex-M4 core features a Floating point unit (FPU) single precision which supports all Arm® single-precision data-processing instructions and data types. It also implements a full set of DSP instructions and a memory protection unit (MPU) which enhances application security. The STM32L476xx devices embed high-speed memories (Flash memory up to 1 Mbyte, up to 128 Kbyte of SRAM), a flexible external memory controller (FSMC) for static memories (for devices with packages of 100 pins and more), a Quad SPI flash memories interface (available on all packages) and an extensive range of enhanced I/Os and peripherals connected to two APB buses, two AHB buses and a 32-bit multi-AHB bus matrix.

2.2.2.3 LSM6DSM

The LSM6DSM is a system-in-package featuring a 3D digital accelerometer and a 3D digital gyroscope performing at 0.65 mA in high-performance mode and enabling always-on low-power features for an optimal motion experience for the consumer. The LSM6DSM supports main OS requirements, offering real, virtual and batch sensors with 4 Kbytes for dynamic data batching.
ST's family of MEMS sensor modules leverages the robust and mature manufacturing processes already used for the production of micromachined accelerometers and gyroscopes. The various sensing elements are manufactured using specialized micromachining processes, while the IC interfaces are developed using CMOS technology that allows the design of a dedicated circuit which is trimmed to better match the

characteristics of the sensing element.
The LSM6DSM has a full-scale acceleration range of ±2/±4/±8/±16 g and an angular rate range of ±125/±245/±500/±1000/±2000 dps. The LSM6DSM fully supports EIS and OIS applications as the module includes a dedicated configurable signal processing path for OIS and auxiliary SPI configurable for both gyroscope and accelerometer.
High robustness to mechanical shock makes the LSM6DSM the preferred choice of system designers for the creation and manufacturing of reliable products.

2.2.2.4 LSM303AGR

The LSM303AGR is an ultra-low-power high-performance system-in-package featuring a 3D digital linear acceleration sensor and a 3D digital magnetic sensor. The device has linear acceleration full scales of ±2g/±4g/±8g/±16g and a magnetic field dynamic range of ±50 gauss.
The LSM303AGR includes an I2C serial bus interface that supports standard, fast mode, fast mode plus, and high-speed (100 kHz, 400 kHz, 1 MHz, and 3.4 MHz) and an SPI serial standard interface. The system can be configured to generate an interrupt signal for free-fall, motion detection and magnetic field detection. The magnetic and accelerometer blocks can be enabled or put into power-down mode separately.

2.2.2.5 LPS22HB

The LPS22HB is an ultra-compact piezoresistive absolute pressure sensor which functions as a digital output barometer. The device comprises a sensing element and an IC interface which communicates through I2C or SPI from the sensing element to the application. The sensing element, which detects absolute pressure, consists of a suspended membrane manufactured using a dedicated process developed by ST. The LPS22HB is available in a full-mold, holed LGA package (HLGA). It is guaranteed to operate over a temperature range extending from -40 °C to +85 °C. The package is holed to allow external pressure to reach the sensing element.
LPS22HB is factory calibrated but a residual offset could be introduced by the soldering process. This offset can be removed with a one-point calibration.

2.2.2.6 MP34DT05-A

The MP34DT05-A is an ultra-compact, low power, omnidirectional, digital MEMS microphone built with a capacitive sensing element and an IC interface. The sensing element, capable of detecting acoustic waves, is manufactured using a specialized silicon micromachining process dedicated to producing audio sensors.
The IC interface is manufactured using a CMOS process that allows designing a dedicated circuit able to provide a digital signal externally in PDM format. The MP34DT05-A is a low-distortion digital microphone with a 64 dB signal-to-noise ratio and –26 dBFS ±3 dB sensitivity.

2.2.2.7 BLUENRG-MS

The BlueNRG-MS is a very low power Bluetooth low energy (BLE) single-mode network processor, compliant with Bluetooth specification v4.1. The BlueNRG-MS supports multiple roles simultaneously and can act at the same time as Bluetooth smart sensor and hub device. The Bluetooth Low Energy stack runs on the embedded ARM Cortex-M0 core. The stack is stored on the on-chip non-volatile Flash memory and can be easily upgraded via SPI.
The device comes pre-programmed with a production-ready stack image(Its version could change at any time without notice). A different or more up-to-date stack image can be downloaded from the ST website and programmed on the device through the ST provided software tools. The BlueNRG-MS allows applications to meet the tight advisable peak current requirements imposed by standard coin cell batteries.
The maximum peak current is only 10 mA at 1 dBm output power. Ultra low-power sleep modes and very short transition times between operating modes allow very low average current consumption, resulting in longer battery life. The BlueNRG-MS offers the option of interfacing with external microcontrollers via SPI transport layer.

2.2.2.8 BALF-NRG-02D3

This device is an ultra-miniature balun which integrates matching network and harmonics filter. Matching impedance has been customized for the BlueNRG transceiver. The BALF-NRG-02D3 uses STMicroelectronics IPD technology on non-conductive glass substrate which optimizes RF performance.

2.3 Firmware Development

The development of the firmware for this board requires a lot of engineers in order to configure all the different modules, starting from the sensing elements to the RF modules.
To write the firmware configuration on the board it is necessary to go through the toolchain that updates the flash memory configurations, so a new application can run on the board. In order to do that, it is necessary to have a platform that enables and makes this toolchain.
For the *Sensortile* from STMicroelectronics is provided the *STMCubeX*, a software platform that has the structure to write the flash memory on the device, in order to run different applications, starting from the C code the configure the modules and the algorithm implemented to preprocess data coming from the sensing elements.
In order to update the flash memory on the device, also some hardware is needed to take the output of the firmware toolchain, the machine code, and put it in the memory of the device.
The device used to do that is the *STM-Nucleo Board*.

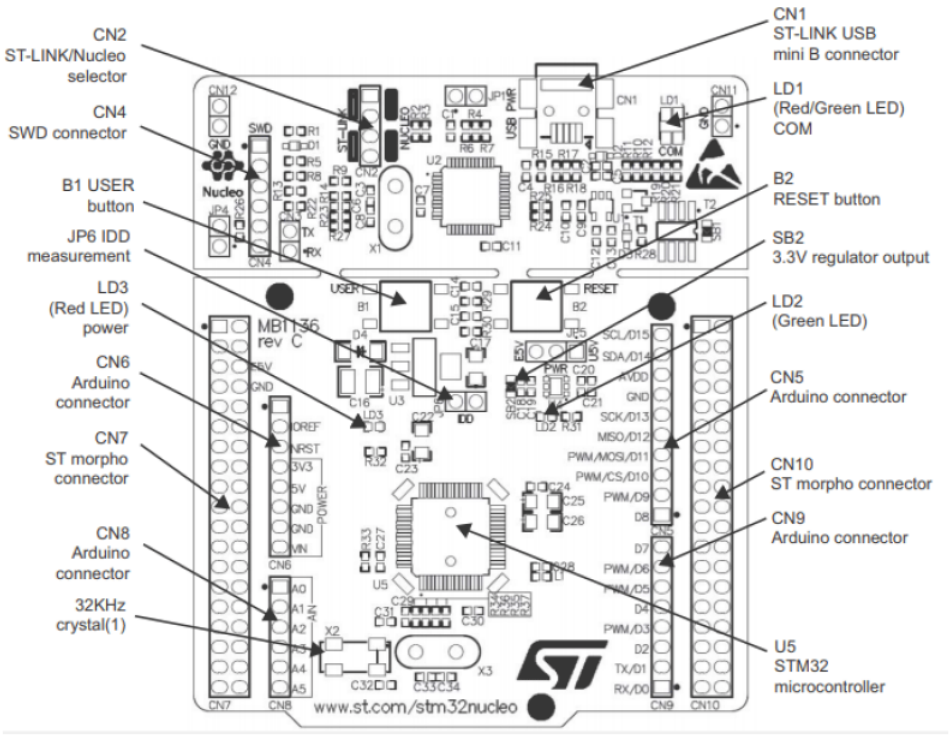


Figure 2.3: STM-Nucleo Board

This board allows the connection on the PC side, where the firmware code is developed and placed on the toolchain. The output of the toolchain is a ".bin" file, a binary file that has to be placed on the flash memory of the device.
In order to connect this board to the Sensortile, some hardware modules are needed:

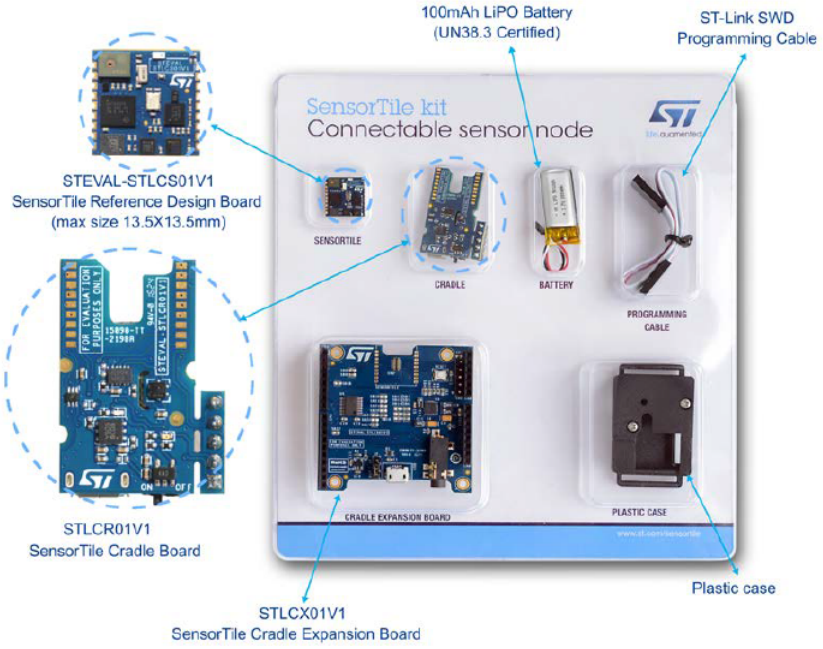


Figure 2.4: Sensortile Kit

The first step that has to be done is to sold the *Sensortile Board* on the cradle board.

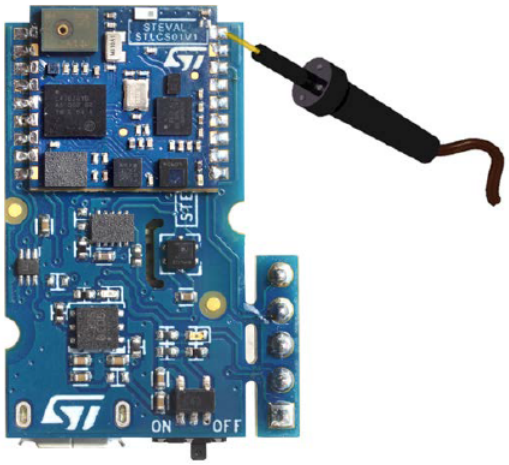


Figure 2.5: Sensortile Board Solded

The second step is to connect the power supply for the device.
The battery LiPO is used to gain power for the board.

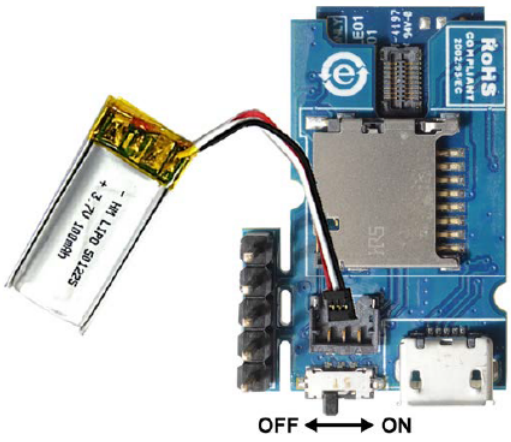


Figure 2.6: Sensortile Board Power Supply

Then, as a final step, the *Sensortile* has to be put into a package.

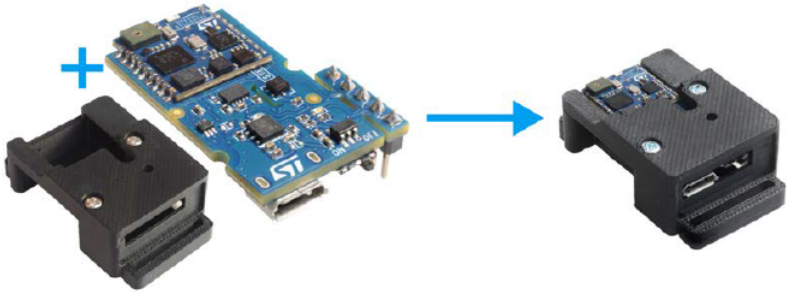


Figure 2.7: Sensortile Board Package

But before it is necessary to write the flash memory of the device to run the developed application. So the STM-Nucleo board and the cradle expansion board have to be attached on the Sensortile Board.
In order to do that, the required Hardware connections are the following:

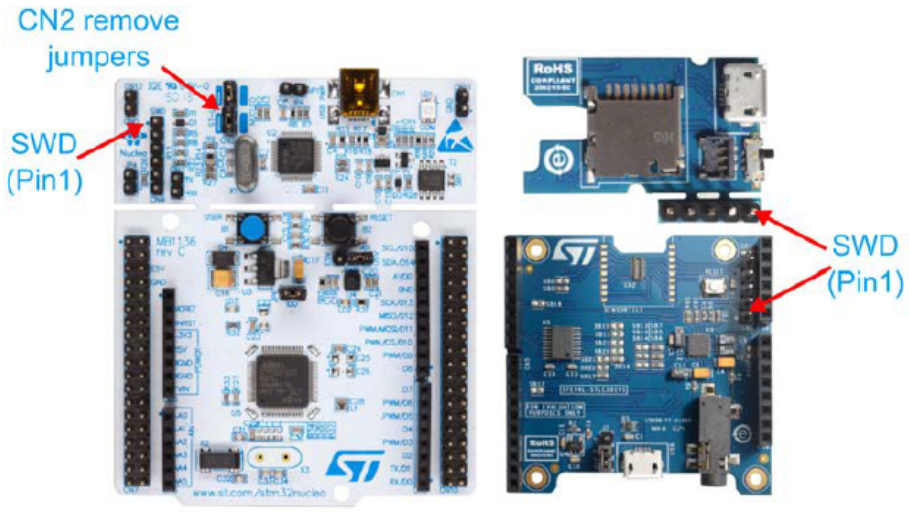


Figure 2.8: Sensortile Board Hardware Connection

As last step is needed to connect the STM-Nucleo Board and the cradle board expansion with the PC, through two USB cables.

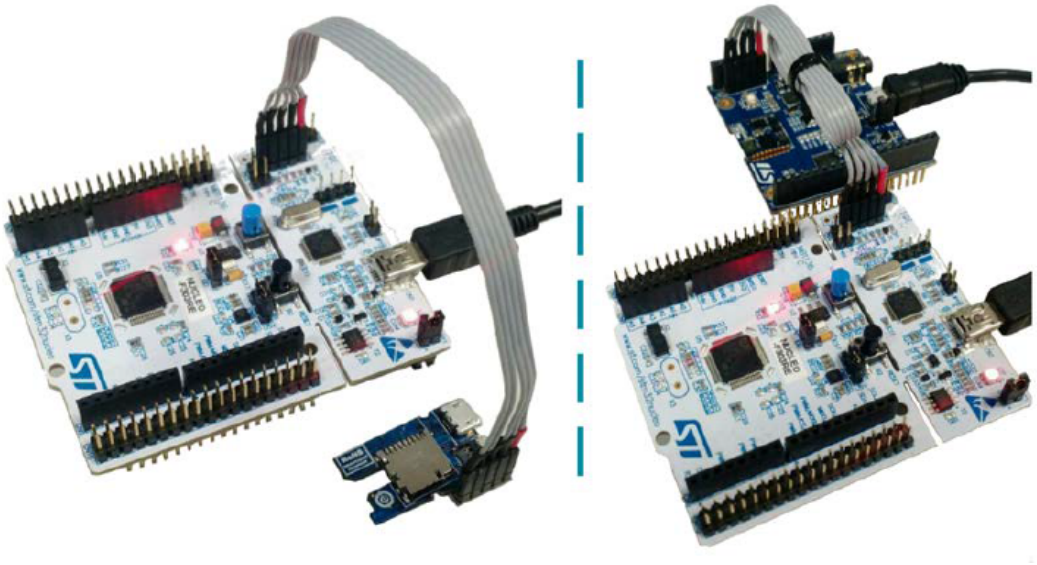


Figure 2.9: Sensortile Board Hardware Connection

2.3.1 Firmware Toolchain

Starting from the C code the firmware toolchain allows to write different applications on the device, with different characteristics in terms of sensing element configuration and RF modules configurations.
C is a compiled language. Its source code is written using any editor of a programmer's choice in the form of a text file, then it has to be compiled into machine code in order to be run by the processor on the device.
C is a general-purpose, procedural computer programming language supporting structured programming, lexical variable scope, and recursion, with a static type system. By design, C provides constructs that map efficiently to typical machine instructions. The flow necessary to change the C code wrote in a file into machine code placed on the memory of the device used to run the application is the following:

2.3.1.1 Preprocessing

The *preprocessor* starts from ".c" files and produces ".c" files. It executes the mechanical operation on the source code, and its directives are given through the "#".
The work that it does is:

- Expands included files (#include)
- Expands macros (#define)
- Removes comments

2.3.1.2 Compiler

It takes the output of the *preprocessor*, which are ".c" files, and it translates this files into assembly language that implements the C code, an intermediate human readable language, specific to the target processor.

- Generates ".s" files wrote in Assembly Language

2.3.1.3 Assembler

The assembler will convert the assembly code into pure binary code or machine code (zeros and ones).

- Generates a ".o" files wrote in sequences of 0's and 1's.

2.3.1.4 Linker

The *linker* merges all the object code from multiple modules into a single one.
If we are using a function from libraries, linker will link our code with that library function code.
In static linking, the linker makes a copy of all used library functions to the executable file.
In dynamic linking, the code is not copied, it is done by just placing the name of the library in the binary file.

- Generates a unique ".bin" file from all ".o" files provided by the *Linker* that can be loaded on the flash memory of the device.

2.3.1.5 Loader

The *Loader* is often a part of the *Linker*, it translates the *abs* file in a form that is understandable for the *Burner*, which is the module responsible for the physical writing of the file on the *Flash Memory* of the device, in order to update the firmware and so to run different programs.

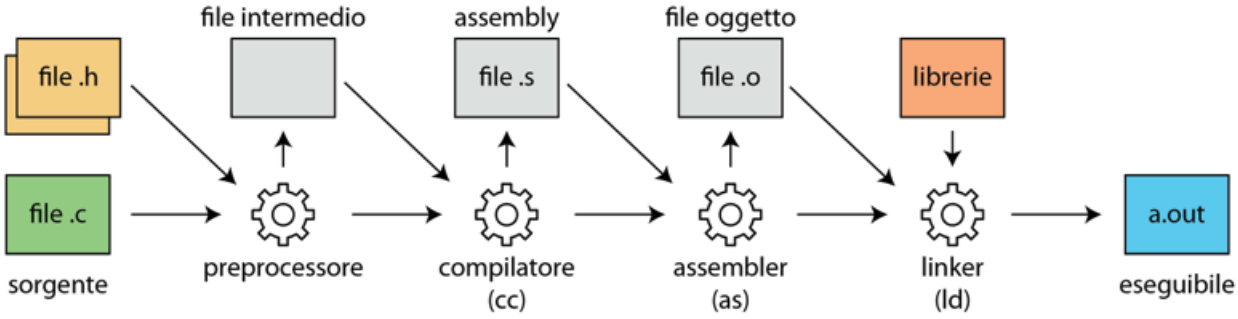


Figure 2.10: Toolchain Steps

2.3.2 Data Acquisition & Sensing Elements

Data acquisition is the process of sampling signals that measure real world physical conditions and converting the results into digital numeric values that can be processed by a computer. Data acquisition systems typically convert analog waveforms into digital values for processing.
In our case of study all data has to be acquired through some sensing elements configured by the firmware. In particular the parameters of the sensors has to be set for the *Inertial Analysis* that is wanted to made.
Some of the main parameter are: the *Sampling Rate*, the *Output Data Rate* and the *Full Scale Values* for all Sensors.
These three parameters have to be set properly on each module for the application.
The sensors configured for our purpose are the following:

2.3.2.1 Accelerometer

Accelerometer measures linear acceleration. It can be used to track the motion, and from its raw data is possible to determine the movement index, the inclination and the vibration measurement of the subject.
MEMS accelerometers embeds several useful features for motion and acceleration detection, including free-fall, wake-up, single/double-tap recognition, activity/inactivity detection and 6D/4D orientation.
ST's accelerometer sensor has advanced power-saving features that make them the ideal choice for ultra-low-power applications. These features include low-power mode, auto wake-up function and FIFO buffer that can be used to store data, thus reducing the host processor loading and system power consumption.

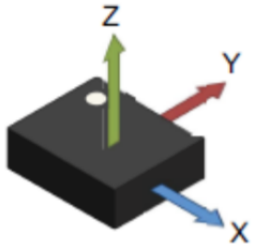


Figure 2.11: Accelerometer

- Output Datat Rate: 100 Hz.
- Full Scale Value: +/- 4 g.

2.3.2.2 Gyroscope

This 3-axis gyroscope has a single sensing structure for motion measurement along all three orthogonal axes, while other solutions on the market rely on two or three independent structures.
ST's solution eliminates any interference between the axes that inherently degrades the output signal, increasing accuracy and reliability of motion-controlled functionalities.
ST's analog and digital gyroscopes offer superior stability over time and temperature, with a resolution lower than 0.01 dps/$\sqrt{}$Hz for zero-rate level. This guarantees the level of accuracy required by the most advanced motion-based applications.

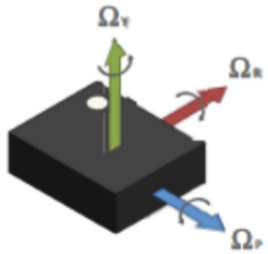


Figure 2.12: Gyroscope

- Output Datat Rate: 100 Hz
- Full Scale Value: +/- 250 dps

2.3.2.3 Magnetometer

The ST Sensortile embeds a 3D digital magnetic sensor with a magnetic field dynamic range of ±50 gauss.

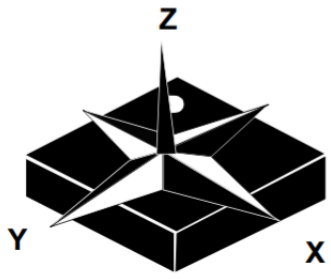


Figure 2.13: Magnetometer

- Output Datat Rate: 100 Hz
- Full Scale Value: +/- 50 gauss.

Integrating all these element an *Inertial Measurement* can be performed strarting from these data.
In order to do that coherently with the analysis to be performed, some parameters have to be changed so that the *Output Data Rate* and the *Full Scale Value* for different sensors is compliant with the specifications required for the *Inertial Analysis*.
It is necessary to discover which are these variables in the code and change them.

2.3.3 Processing of data on board

It is necessary to perform as less as possible computation in order to save battery charge because of the long time needed to store data, around 8 hours.
Also if others sensing elements will be integrated, the idea is to simply acquire raw data, and the computation of these data is moved on the server side.
The analysis of data is moved on the Server side.

2.3.4 Storage of Data

The storage of samples during the data acquisition its done in a micro-SD, which can be made of dozens of GBytes.
So all raw data can be stored without memory problems in terms of space needed to store all data.

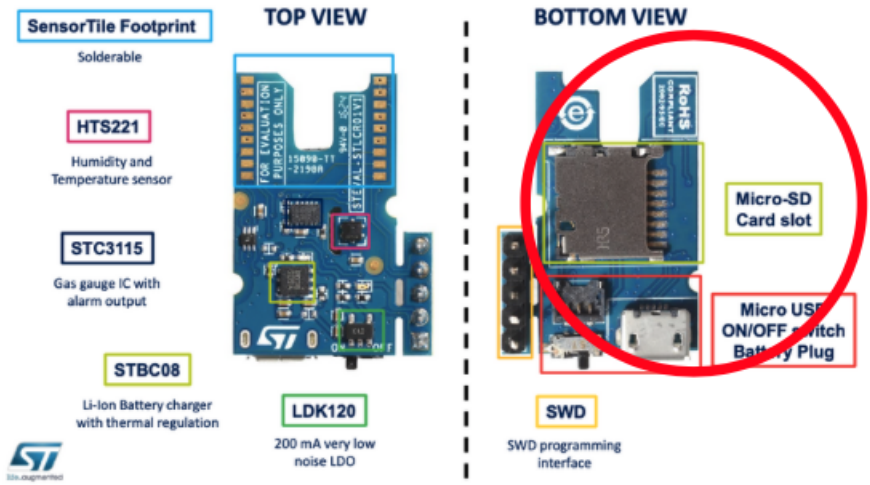


Figure 2.14: Micro-SD Slot

2.3.5 Communication of data through BLE protocol

Bluetooth is a radio technology, supporting communication of device in short distance, and making wireless information transfer between numerous devices possible. The Bluetooth has been used in a series of technologies, methods and theories for hardware and software designs. For example, wireless communication and technologies in network, engineering and software dependability theory, protocol testing technology, standard describing language, built-in RTOS, cross-platform development and graphical user interfaces technology, interface technology for software and hardware, and CMOS chips integration technology.
Because of the small size and the low power, the application of Bluetooth technology is more than a computer's peripheral device. It can be integrated inside of any digital device, especially for micro devices and portable devices, which do not require high quality on transfer speed.
For this thesis this protocol is used to transfer data from the Sensortile, which stores raw data in the micro-SD card, to the personal Smartphone.
The transmission between the Smartphone and the server station has to be studied in deeper because of the very sensitive data stored, and is not treated in this work.

Figure 2.15: Micro-SD Slot

2.4 Analysis of Data

2.4.1 Type of Analysis

- Statistical Analysis
  Statistical analysis means investigating trends, patterns, and relationships using quantitative data. It is an important research tool used by scientists, governments, businesses, and other organizations.
  To draw valid conclusions, statistical analysis requires careful planning from the very start of the research process. You need to specify your hypotheses and make decisions about your research design, sample size, and sampling procedure.
  After collecting data from your sample, you can organize and summarize the data using descriptive statistics. Then, you can use inferential statistics to formally test hypotheses and make estimates about the population. Finally, you can interpret and generalize your findings.
  The *Statistical Analysis Process* starts with the research design, which is your overall strategy for data collection and analysis. It determines the statistical tests you can use to test your hypothesis later on.
  Is necessary to decide whether your research will use a descriptive, correlational, or experimental design. Experiments directly influence variables, whereas descriptive and correlational studies only measure variables.
  Your research design also concerns whether you'll compare participants at the group level or individual level, or both.
  When planning a research design, you should operationalize your variables and decide exactly how you will measure them.
  For statistical analysis, it's important to consider the level of measurement of your variables, which tells you what kind of data they contain:

  – Categorical data represents groupings. These may be nominal (e.g., gender) or ordinal (e.g. level of language ability).
  – Quantitative data represents amounts. These may be on an interval scale (e.g. test score) or a ratio scale (e.g. age).

  Many variables can be measured at different levels of precision. For example, age data can be quantitative (8 years old) or categorical (young). If a variable is coded numerically (e.g., level of agreement from 1–5), it doesn't automatically mean that it's quantitative instead of categorical.
  Identifying the measurement level is important for choosing appropriate statistics and hypothesis tests.

For example, you can calculate a mean score with quantitative data, but not with categorical data.
In a research study, along with measures of your variables of interest, you'll often collect data on relevant participant characteristics.
The next step is collecting data.
In most cases, it's too difficult or expensive to collect data from every member of the population you're interested in studying. Instead, you'll collect data from a sample.
Statistical analysis allows you to apply your findings beyond your own sample as long as you use appropriate sampling procedures. You should aim for a sample that is representative of the population.
With data collected the next step is the *Preprocessing* phase, which is a crucial step that has a huge impact on the performance of the analysis process. It refers to manipulate data or drop them, in order to have a clean dataset.
After that the *Analysis Process* can be performed, it consists in applying different algorithms to data in order to extract relevant features for the final goal of the application.
Finally the results can be consulted by the experts of the application.

Figure 2.16: Statistical Analysis

- AI Analysis
  To apply AI analysis means to find correlations on data using algorithms that improve their performance at some task with experience. It can be seen as a system that can continuously improve it self by learning from analytical observation of the input data.
  The main focus is on the Machine Learning (ML) algorithms which can learn from the input applied to it; they are subdivided in three kind of learning algorithms: supervised learning, unsupervised learning and reinforcement learning.
  For this application the analysis has the goal of classifying correctly the different sleep stages. It can be seen as a multi label classification problem, which can be approached with classification and clustering algorithms.
  If the dataset is unlabeled is possible to try with clustering algorithms, but the performance of the classification can be better then the clustering approach, because you are applying also the label of each sample to the algorithm.
  For this work the ML algorithms are used to analyze the datatset. The flow of this analysis process is detailed in the next section.

2.4.2 Data Analysis Process

Data Analysis is the process of collecting, transforming, cleaning, and modeling data with the goal of discovering the required information. The obtained results are then reported, suggesting conclusions, and supporting decision-making. Data visualization is at times used to portray the data for the ease of discovering the useful patterns in the data. The terms Data Modeling and Data Analysis mean the same.

2.4.2.1 Data Cleaning

Clear Dataset from NaN values

The first thing to do is clear the dataset from NaN values, which are missing data.
These NaN values might be due to different problems during the acquisition of the physical quantities, for instance when the physical quantity exceeds the dynamic range allowed to be sensed from the sensing element, or data lost when transmitted to the database.
So these missing data on the dataset need to be cleaned.
The type of *Data Cleaning* method has to be comply with the type of data (boolean, integer, float, ...).

Balance the Dataset

After the *Cleaning* step, which is really important for the other steps in terms of their performance, in particular for the analysis step, it is necessary to split the dataset into traning set and test set, in order to train the model that will be implemented, and test it on the test set.

There is not a formal rule to perform this split, but it is possible to say that there are two competing concerns: with less training data, the parameter estimates have greater variance. With less testing data, the performance statistic will have greater variance. Broadly speaking one should be concerned with dividing data such that neither variance is too high, which has more to do with the absolute number of instances in each category rather than the percentage.

So it is necessary to make a decision: one possible solution is that the training set comprises the 80/90% of the total data and the test set the remaining 20/10%.

At this point there is one main problem: is the training set balanced??

The number of samples belonging to different classes has to be the same in order to train properly the model that will be used, and this concern both *Clustering* and *Classification*.

An over-sampling strategy is needed because of the few samples on the healthy class.

The strategy that will be used is the so called SMOTE oversampling.

This technique is able to identify a geometrical region in which the samples are placed, and creates new samples inside that region:

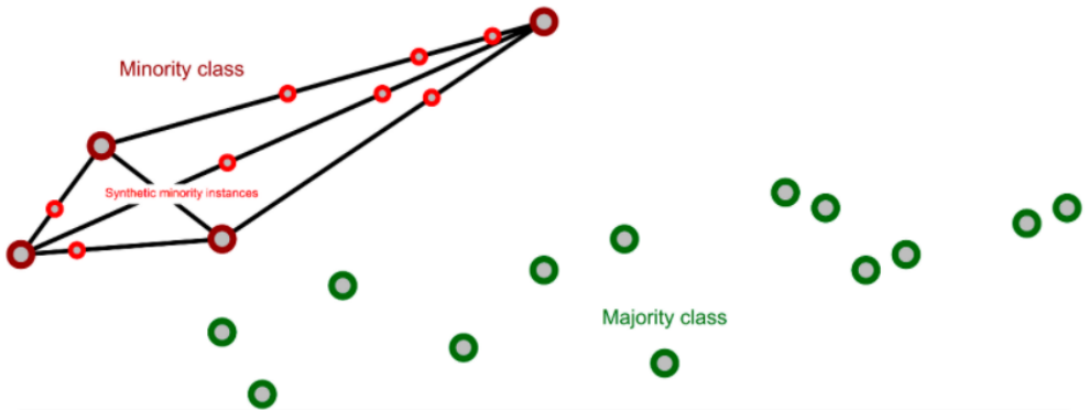


Figure 2.17: SMOTE Oversampling

A further improvement that can be done to balance the dataset, is to implement an *Autoencoder* to encode the overall samples and use the most hidden layer that encode the input, to generate new samples.
The Autoencoder is capable of creating sparse representations of the input data, it can be a better way to oversample the input dataset for balancing the data.

2.4.2.2 Data Preprocessing

In any Machine Learning process, Data Preprocessing is the step in which the data gets transformed, or Encoded, to bring it to such a state that the machine can easily parse it. In other words, the features of the data can be easily interpreted by the algorithm. Data preprocessing is a required first step before any machine learning method can be applied, because the algorithms learn from the data and the learning outcome for problem solving heavily depends on the proper data needed to solve a particular problem, which are called features.
These features are key for learning and understanding, and therefore, machine learning is often considered as feature engineering. Data preprocessing, however, inflicts a heavy danger; for example, during the preprocessing, data can be inadvertently modified; for example, "interesting" data may be removed.
Consequently, for discovery purposes, it would be wise to have a look at the original raw data first and maybe do a comparison between nonprocessed and preprocessed data.

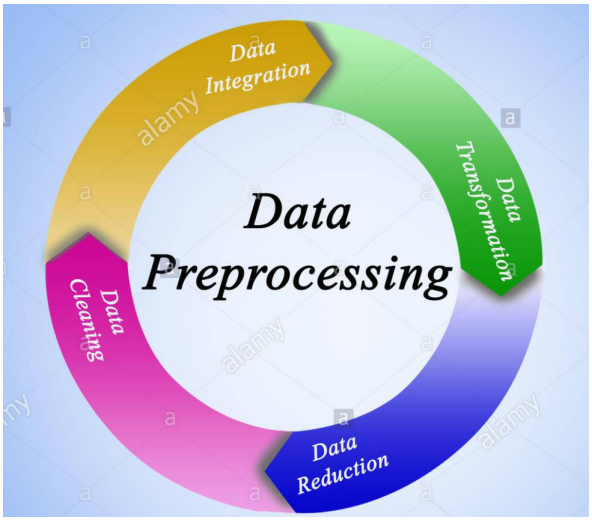


Figure 2.18: Data Preprocessing

Data Scaling

First of all, as preprocessing step, one may *scale* the values of the dataset, in order to properly train the model that will be used to make the results of the data analysis.
Feature scaling is an important step during data pre-processing to standardize the independent features present in the dataset.
Standardizing means to scale the features in order to bring them to the same range. There are multiple techniques to perform feature scaling. But, first, let's understand why is it important to do so.
In a general scenario, every feature in the dataset has some units and magnitude; the machine learning model will give high importance to those that have high magnitude and low importance to features that have low magnitude, regardless of the unit of the values.
Types of data scaling:

- Standard Scaler
  In this approach, all the features are rescaled to a similar scale centring the feature array at 0 with a standard deviation of 1. In the case of outliers, this scaler technique will be affected. Hence, it is used when the features are normally distributed.

$$x_{std\_scaled} = x_i - mean(x)/std(x)$$

- Min Max Scaler
  This estimator scales each feature array individually such that it is in the given range, for instance between zero and one. This technique is mainly used in deep learning and also when the distribution is not Gaussian. This scaler is also sensitive to outliers.

$$x_{min\_max\_scaled} = x_i - min(x)/max(x) - min(x)$$

- Gaussian Transformation
  Gaussian distribution is nothing but normal distribution. In case our features are not normally distributed, we can apply some transformations to make them normally distributed.
  There are different transformation:
  - Logarithmic Transformation
  - Reciprocal Transformation
  - Square Root transformation
  - Exponential Transformation

Dimentionality Reduction

Dimensionality reduction is a popular method in machine learning commonly used by data scientists.

Given a dataset of various features, you can reduce the number of features through feature engineering and feature selection. This consists in it's self is intuitively reducing the dimensions of the original dataset you were working with.

For instance, assuming you have a certain number of features that you can combine a set of features without losing information about those by doing some arithmetic, then the new feature would replace the pairs of features it was produced from. The process of creating new features through preprocessing is known as feature engineering, and the process of selecting specific features for the purposes of training a model is known as feature selection.

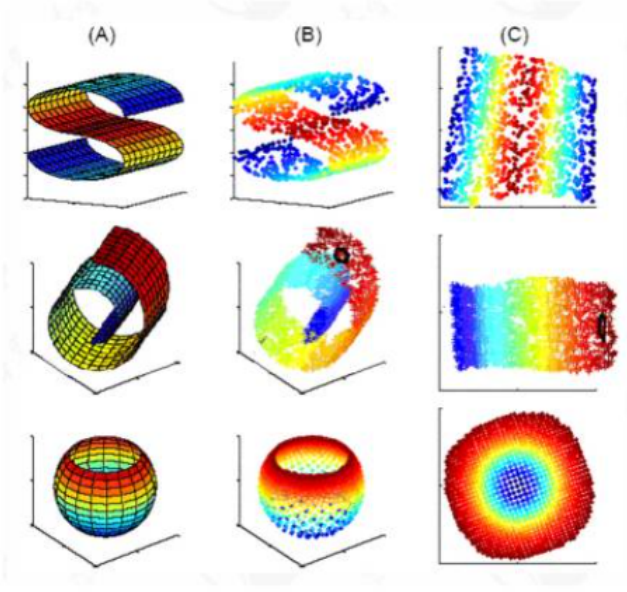


Figure 2.19: Dimentionality Reduction

Another technique is the *Principal Component Analysis* (PCA).

PCA is a highly used unsupervised learning technique to reduce the dimension of a large dataset. It transforms the large set of variables into smaller components which contains the majority of the information in the large one. Reducing the size of the dataset could naturally result in loss of information and impact on the accuracy of the model. However, this downside is offset by the ease of use for exploration, visualization

and analysis purposes. Let's make an example.
PCA aims to create new characteristics which summarize the initial characteristics. Through finding linear combinations of the old characteristics, PCA can construct new characteristics whilst trying to minimize information loss.

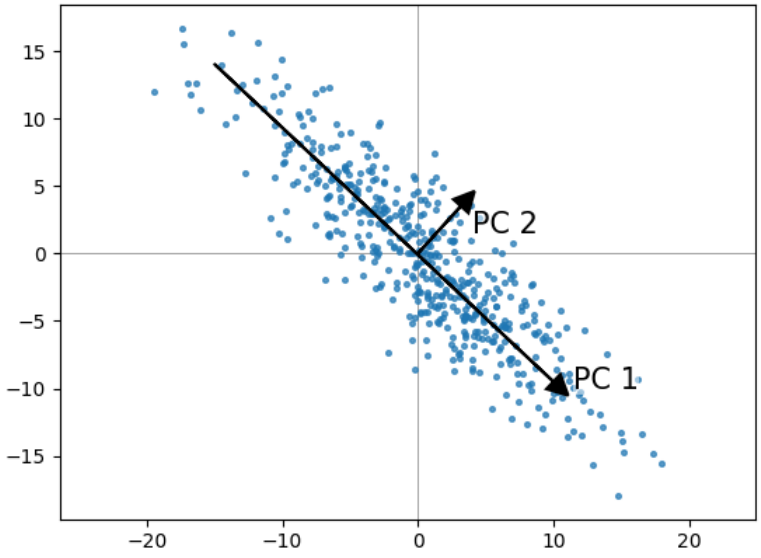


Figure 2.20: Principal Component Analysis

2.4.2.3 Data Analysis

Clustering

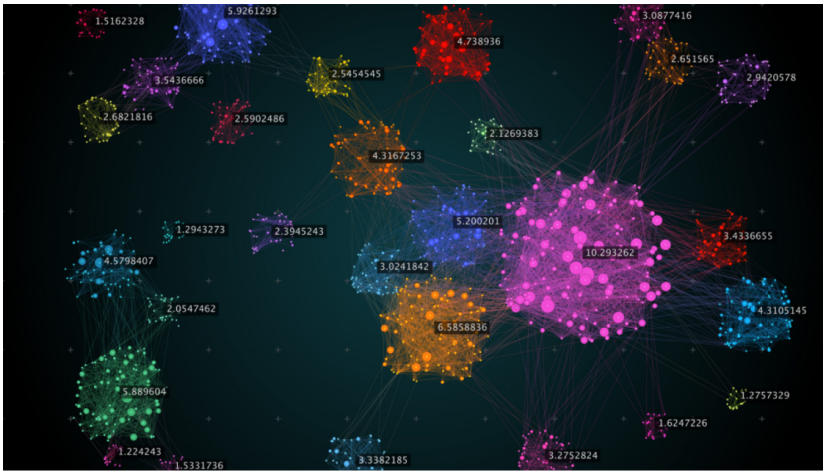


Figure 2.21: Clustering

Clustering is the assignment of objects to homogeneous groups (called clusters) ensuring heterogeneity among different clusters.
Clustering is considered an unsupervised task as it aims to describe the hidden structure of the objects.
A cluster refers to a collection of data points aggregated together because of certain similarities.
The number of final (or desired) clusters is determined by the number of centroids.
Each cluster has a centroid, which is the imaginary or real location representing the center of the cluster.
The first step of dividing objects into clusters is to define the distance between the different objects. Defining an adequate distance measure is crucial for the success of the clustering process.
Then, each data point is assigned to a cluster on the basis of some distance-based metrics..
The instances are partitioned into a number of classes based on the:

- Maximization of similarity of instances of the same cluster.
- Minimization of similarity of instances of different clusters.

There are different cluster strategies in which the model can be defined, looking at different characteristics.
Clustering Strategies:

- Flat (e.g. k-means)
  – Iteratively re-assign points to the nearest cluster center.
- Hierarchical clustering
  – Iteratively merge or split the clusters.
- Density based clustering (e.g. mean shift)
  – Estimate modes of probability density function.

Classification

Classification is the process of predicting the class of given data points. Classes can be labels or categories. Classification consists in approximating a mapping function (f) from input variables (X) to discrete output variables (y).
Classification encompassess all methods of supervised learning; that is when the targets are provided along with the training data.

- Classification Algorithms
  Classification methods comprise various algorithms; however, their performance depends on the application and on the characteristics of the dataset, so it is not possible to define the best method a-priori.
  - *Decision Tree*
    Decision tree builds classification or regression models in the form of a tree structure. It utilizes an if-then rule set which is mutually exclusive and exhaustive for classification. The rules are learned sequentially using the training data one at a time. Each time a rule is learned, the tuples covered by the rules are removed. This process is continued on the training set until meeting a termination condition.

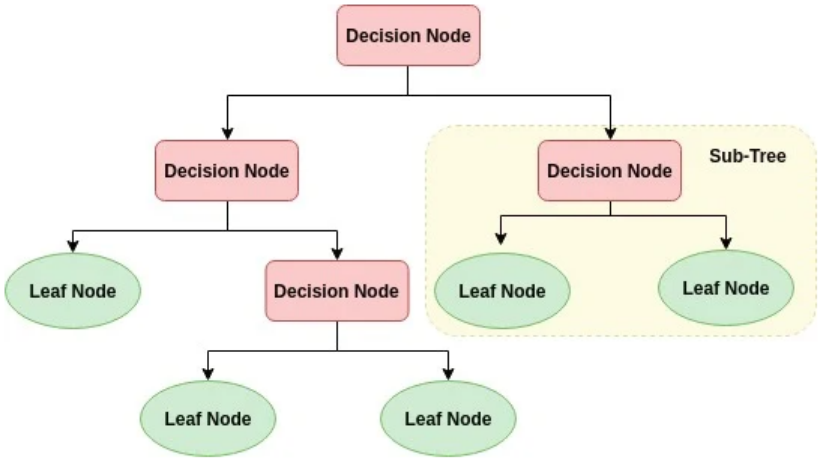


Figure 2.22: Decision Tree

The tree is constructed in a top-down recursive divide-and-conquer manner. All the attributes should be categorical. Otherwise, they should be discretized in advance. Attributes in the top of the tree have more impact towards in the classification and they are identified using the information gain concept. A decision tree can be easily over-fitted generating too many branches and may reflect anomalies due to noise or outliers. An over-fitted model has a very poor performance on the unseen data even though it gives an impressive performance on training data. This can be avoided by pre-pruning which halts tree construction early or post-pruning which removes branches from the fully grown tree.

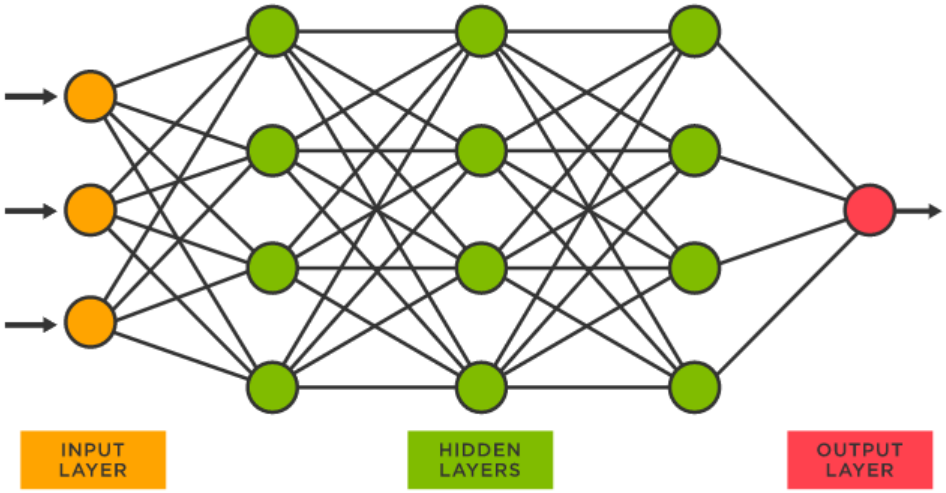
– *Naive Bayes*
Naive Bayes is a probabilistic classifier inspired by the Bayes theorem under a simple assumption which is the attributes are conditionally independent.

$$P(\mathbf{X} \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i) = P(x_1 \mid C_i) \times P(x_2 \mid C_i) \times \ldots \times P(x_n \mid C_i)$$

The classification is conducted by deriving the maximum posterior which is the maximal P(Ci|X) with the above assumption applying to Bayes theorem. This assumption greatly reduces the computational cost by only counting the class distribution. Even though the assumption is not valid in most

cases since the attributes are dependent, surprisingly Naive Bayes has able to perform impressively.
Naive Bayes is a very simple algorithm to implement and good results have obtained in most cases. It can be easily scalable to larger datasets since it takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers. Naive Bayes can suffer from a problem called the zero probability problem. When the conditional probability is zero for a particular attribute, it fails to give a valid prediction. This may require fixing, which could be carried out by adopting a Laplacian estimator.

– *Artificial Neural Network*
Artificial Neural Networks are a set of connected input/output units where each connection has a weight associated with it. It was started by psychologists and neurobiologists to develop and test computational analogs of neurons. During the learning phase, the network learns by adjusting the weights so as to be able to predict the correct class label of the input tuples.



There are many network architectures available now like Feed-forward, Convolutional, Recurrent etc. The appropriate architecture depends on the application of the model. For most cases feed-forward models give reasonably accurate results and especially for image processing applications, convolutional networks perform better.

There can be multiple hidden layers in the model depending on the complexity of the function which is going to be mapped by the model. Having more hidden layers will enable to model complex relationships such as deep neural networks.
However, when there are many hidden layers, it takes a lot of time to train and adjust wights. The other disadvantage of is the poor interpretability of model compared to other models like Decision Trees due to the unknown symbolic meaning behind the learned weights.
But Artificial Neural Networks have performed impressively in most of the real world applications. It is high tolerance to noisy data and able to classify untrained patterns. Usually, Artificial Neural Networks perform better with continuous-valued inputs and outputs.

2.5 Store data on a Database

2.5.1 SQL Database

The Structured Query Language (SQL) is the most extensively used database language. SQL is composed of a data definition language (DDL), which allows the specification of database schemas.
A data manipulation language (DML), which supports operations to retrieve, store, modify and delete data; and a data control language (DCL), which enables database administrators to configure security access to databases. Among the most important reasons for SQL's wide adoption are that:

- it is primarily a declarative language, that is, it specifies the logic of a program (what needs to be done) instead of the control flow (how to do it).
- it is relatively simple to learn and understand because it is declarative and uses English statements.
- it is a standard of the American National Standards Institute (ANSI) and the International Organization for Standardization (ISO).
- it is, to some extent, portable among different database systems.

Even though SQL was initially proposed for traditional relational databases, it has also been found to be an effective language in several new types of database systems, particularly Big Data management systems (BDMSs) that process large, fast and diverse data. While BDMSs have significantly changed the database landscape and traditional RDBMs represent now only a small fraction of it, most books and database courses only present SQL in the context of RDBMs.
Basically SQL extracts record sets from huge databases based on a relational algebra. SELECT is the core SQL, endowed with powerful clauses for filtering records, columns/attributes, computation, grouping, etc.
The immense popularity of SQL (Michael Stonebraker once called SQL the intergalactic dataspeak language) is due mainly to its high level syntax (no programming is necessary for most of the queries) and also to its implementation in all types DataBase Management Systems, from desktop (Access) to open-source (MySQL, PostgreSQL) and commercial (Oracle, IBM DB2, Microsoft SQL Servers) ones.
The broad adoption was facilitated by the standardization of SQL by ISO with ANSI and various national agencies. First SQL standard was published in 1986 (ANSI) and 1989 (ISO), and then in 1992, 1999, 2003, 2008 and 2011.
As pointed out in previous section, the result of SQL queries (SELECT commands)

can be saved/stored inside the database (mainly as table or view) but also is prone to be exported from the DBMS to various targets and formats, i.e. another database, ExcelCSV file, text file, HTML, ODBCJDBC data source, etc. But SELECT commands do not merely extract and filter data from the database. Its various clauses can do various processing tasks for all the result rows or for groups or rows (GROUP BY and HAVING clauses).
Starting with the first standard (1986/1989), all SQL dialects have implemented the basic statistical functions called (statistical) aggregate functions with self-descriptive names: SUM, COUNT, AVG, MIN, MAX. Since 1999 one of the most important target of SQL standards has been data analysis, mainly through OLAP (On Line Analytical Processing) features (sometimes also called window functions). There are some OLAP differences among dialects.

- Store raw data and processed data in a SQL Data Structure
  - Raw data, for further testing on results or other studies
  - Processed data, these data are like a rows in a dataframe, with the relative value of the different features

The general schema for storage and access to data:

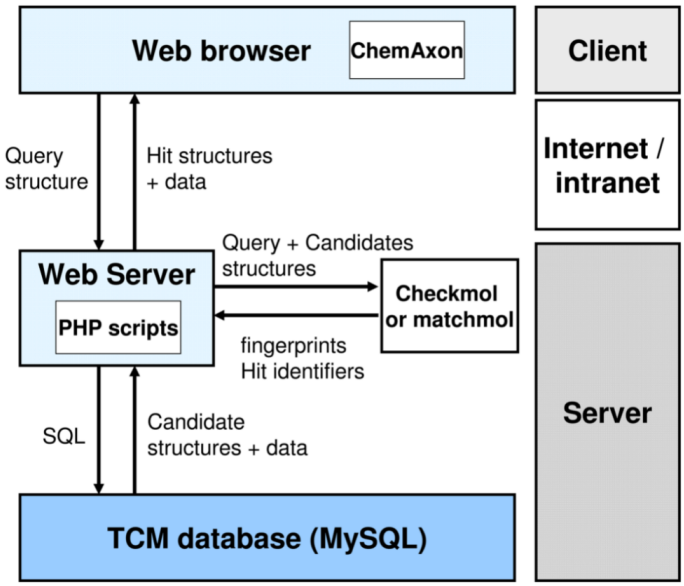


Figure 2.23: Schema for Data Storage

2.6 GUI Interface

A graphic user interface (GUI) is developed in order to easily get access to raw data by pathologist.
This interface is implemented in Python, through the usage of Tkinter package, with is a set of packages that helps in implementing a GUI. It is necessary to develop the front-end of the GUI. To do that different fields are defined, which can be made of entries, where information can be written, labels, that show information through the GUI, and buttons, to trigger the scripts and upload or get information.
This interface can communicate with the SQL database through the *sqlite3* package of Python. This is a really important feature because it implies that the database can be uploaded in a really simple way. At the beginning is necessary to populate the database with raw data coming from patients. The GUI is configured in a way in which is simple to add and remove patients, to upload raw data on the database, to upload the analysis results, and finally also to effectively run the analysis process.
It is possible to directly write on the boxes all the information needed, but it's necessary to take into account that this program might be on wrong hands. To avoid possible security problems through this script, a white list of character is defined in order to cut off a lot of cybersecurity problems that may happen.
One of the analysis that can be done through this interface is the statistical analysis. In this analysis the goal is to show the most relevant feature for a pathologist. The raw data are split into 30 seconds window, and different metrics are computed on each of these windows. In this way the pathologists can have all the wanted parameter to analyze data also by their self, looking at the metrics values.
They have to be helped by automatic tools that are able to find more tiny correlation among data. To do that, machine learning algorithms have to be used, and the interface is structure in a way in which is really simple to integrate new analysis process or new metrics.
The already available features are the following:

- Connection with Database
- Tables structure of the Database
- Add, remove and upload patients
- Add, remove and upload patients raw data
- Add, remove and upload patients results analysis
- Main metrics can be plotted

2.7 Heterogenous Data Integration

In our case of study the *Sensortile Board* is used to keep track of the motion parameters and the temperature of the patient.
The idea is to place three of them on the body: one on one side of the pelvis, one on one hand and one on one foot.
These three regions give us a lot of information on the overall movement index of the patient.
To keep track of the *heart rate* and the *blood oxygen* in possible to use different commercial devices, so that heterougenous data can be integrated.
Data integration consists in providing a uniform view of a set of heterogeneous data sources. This allows users to define their queries without any knowledge on the heterogeneous sources.
Data integration systems use the mediation architecture to provide integrated access to multiple data sources. A mediator is a software device that supports a mediation schema which captures user requirements, and a set of mappings between the mediation schema and the distributed sources. Mediation systems can be classified according to the approach used to define the mappings between the data sources and the global schema.
In our case raw data coming from different sensing elements can have different output data rate, so is necessary to ensure coherence between these data.
The idea is to add raw data for the sensors with less output data rate, coping its value for the required time, which is n times in order to have the same number of data in the same period window between two samples.
In general adding smart devices which are able to sense different physical quantities, and integrating all these informations, the predictive models can perform better.

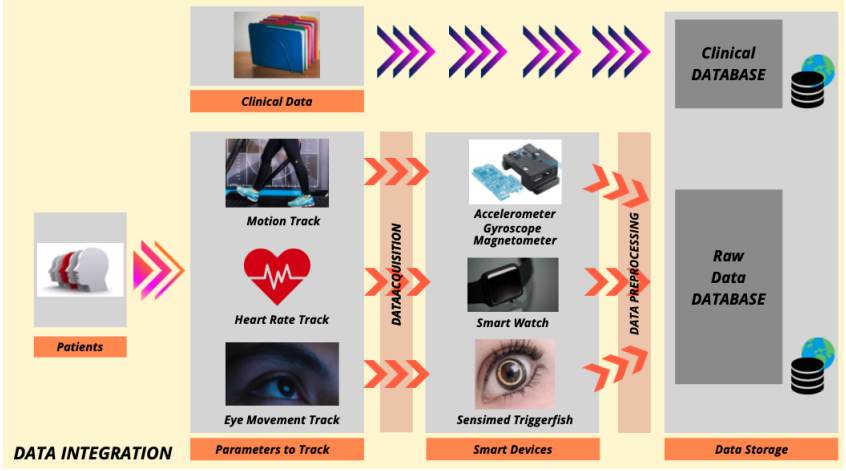


Figure 2.24: Data Integration

2.8 Case of Study

For this thesis was not possible to track directly some patients in order to acquire raw data because of the Coronavirus pandemic.
So in this thesis are used data from online dataset (https://physionet.org/content/sleep-accel/1.0.0/).
The aim is to analyze heterogeneous data coming from accelerometer and heart rate MEMS, which are acquire simultaneously with the PSG exam, so that the results coming from this exam can be used as labels to train models with raw data from sensors as input. If the amount of datasets acquired in this way increases, the models can be deeper and can extract much more features to increase performances.
Potentially if some models gives approximately a very high score in terms of performances, it can be used in inference with raw data coming from smart device instead of the PSG exam, or almost as first check, avoiding the main problems of this technology.

2.8.1 Data Collection

Initially 39 subjects were recruited to participate for this study. An exclusion criteria was adopted in order to make sure that participants did not have a known diagnosis of the following: insomnia, parasomnias, restless leg syndrome, sleep related breathing disorder, cardiovascular disease, heart disease and cardiovascular disease, or other disease that can be neurologically significant and can impair on the final results.
After that the enrolled participants were provided with an Apple Watch befere the PSG exam, so during the entire PSG recording, the Apple Watch has enabled accelerometer sensor and Photoplethysmography (PPG) sensor to track motion and heart rate with the smart device.
The PSG exam was conducted in accordance with the American Academy of Sleep Medicine (AASM). Bilateral frontal, central and occipital electroenceph- alogram (EEG) recorded with use of the International 10–20 system of electrode placement, bilateral electrooculogram (EOG) recorded from the supraorbital and infraorbital ridges, chin electromyogram (EMG), thoracic and abdominal respiratory in- ductance plethysmography (RIP) belts, snore microphone, pulse oximetry, and electrocardiogram (ECG) with use of two leads were recorded.
The PSG exam results was analyzed by the expert team which labeled the different sleep phases. So that these infformation can be used as label in the classification process.
In cases where the battery on the Apple Watch failed before the sleep opportunity ended, the data was cropped to include only those time points for which valid data existed.

2.8.2 Data Preprocessing

It is necessary to manipulate and compute raw data in order to extract relevant features for the analysis process. The chosen features are three:

- Motion: activity count from raw data of accelerometer
- Heart Rate: directly from the PPG
- Clock Proxy: term representing simulated input to sleep from the circadian clock

Acceleration was returned from the Apple Watch as three vectors representing acceleration in the x, y, and z directions, and a fourth, representing the timestamp of the measurement in seconds since January 1, 1970 (UNIX or epoch time), and the acceleration in each direction was returned in units of g.
Heart rate was measured by PPG from the Apple Watch and returned in beats per minute sampled every several seconds.
This signal was interpolated to have a value for every 1 second, smoothed and filtered to amplify periods of high change by convolving with a difference of Gaussians filter (1 = 120 seconds, 2 = 600 seconds).
Already at this point it's necessary to think about how to integrate information that have different output data rate and result as different number of samples.
The solution is given by expanding raw data of the sensors which have less number of samples. Increasing the number of samples by copying the values for all the required times, in order to have on the same window of interest the same number of samples.
By "clock proxy," we refer to a feature meant to approximate the changing drive of the circadian clock to sleep over the course of the night.
The clock-proxy feature was determined by two separate ways. The first way was to use a fixed cosine wave, shifted relative to the time of recording start, which rose and fell over the course of the night. This way of computing the clock proxy term is attractive because it only requires the time of recording as an input.
It is possible to add new features and put them in input of the analysis process, so that different results may appear, and can be better or not, this has to be tried for further research on this field of applications.
The Python scripts that implements the preprocessing step can be easily updated because of its Object Oriented structure, that makes simple the integration of new analysis ideas.

2.8.3 Data Analysis

After the preprocessing step of raw data, and the feature extraction, the models that will be used to find correlations on feature are the following:

- Logistic Regression
- K-nearest neighbors
- Random Forest
- Neural Network

The model are defined from pre-builded tools provided by scikit learn for Python.
The code that implements the preprocessing and the analysis step is available at https://github.com/ojwalch/sleep_classifiers.
Initially, all training and testing was done within the Apple Watch dataset. Classification of sleep stage (either sleep/wake or wake/NREM/REM) by each of the models considered was compared to PSG in an epoch-by-epoch analysis.
Models were trained and tested using both Monte Carlo cross-validation and leave-one-out cross-validation. For Monte Carlo cross-validation with sleep/wake classification, the dataset was randomly split 50 times into a training set (approximately 70% of the subjects) and a testing set (approx. 30%), and for wake/NREM/REM classification, the dataset was randomly split 20 times at the same training and testing proportions. In the leave-one-out cross-validation, a single subject was held out for testing, and the model was trained on the remaining subjects. No samples in the training set were ever used in the corresponding testing set, nor were samples from a single subject ever simultaneously used in both the training and testing sets. Parameters were tuned for each training dataset to minimize the risk of overfitting.

2.8.4 Results of Analysis Process

Sleep/Wake up Classification
In the case of binary sleep/wake classification, local heart rate standard deviation by itself (without motion) was consistently the lowest performing feature set for the classifiers, scoring roughly 24%–33% of wake epochs correctly (specificity) when the fraction of sleep epochs scored correctly (sensitivity) was fixed at 90% across classifiers. The motion-only feature set identified 48%–55% of wake epochs correctly when the fraction of correct sleep epochs was fixed at 90%. Combining motion and heart yielded few improvements to sleep/wake classification over motion-only for binary sleep/wake classification (adding only roughly 3% to the fraction of wake scored correctly in k-nearest neighbors at the 95% threshold for the fraction of sleep epochs scored correctly). The inclusion of the clock proxy improved the fraction of wake epochs scored correctly by

about 14% (when the fraction of sleep epochs scored correctly was fixed at 90%) when added to motion and heart rate in both the random forest and neural net classifiers.In the case of binary sleep/wake classification, local heart rate standard deviation by itself (without motion) was consistently the lowest performing feature set for the classifiers, scoring roughly 24%–33% of wake epochs correctly (specificity) when the fraction of sleep epochs scored correctly (sensitivity) was fixed at 90% across classifiers.
The motion-only feature set identified 48%–55% of wake epochs correctly when the fraction of correct sleep epochs was fixed at 90%. Combining motion and heart yielded few improvements to sleep/wake classification over motion-only for binary sleep/wake classification (adding only roughly 3% to the fraction of wake scored correctly in k-nearest neighbors at the 95% threshold for the fraction of sleep epochs scored correctly). The inclusion of the clock proxy improved the fraction of wake epochs scored correctly by about 14% (when the fraction of sleep epochs scored correctly was fixed at 90%) when added to motion and heart rate in both the random forest and neural net classifiers.

Wake/NREM/REM Classification

Two different approaches were employed for the analysis of the wake/NREM/REM classifier performance: traditional ROC curves, and one versus rest ROC curves. Typically, ROC curves are generated for binary classification problems. In cases where there is more than one class, as in wake/NREM/REM classification, the definition of "true positive" on the y-axis is ambiguous; therefore, one versus rest ROC curves for each class were also used; that is, wake versus not wake, REM versus not REM, and NREM versus not NREM. This reduces the classification problem to a binary one.
Additional ROC curves summarize the performance in all three classes by replacing "true positive" with the accuracy where REM and NREM performance is (approximately) equal. These multi-class staging ROC curves were generated by applying two thresholds to the probabilities returned from the classifier.
The results of the classification discussed are show below:

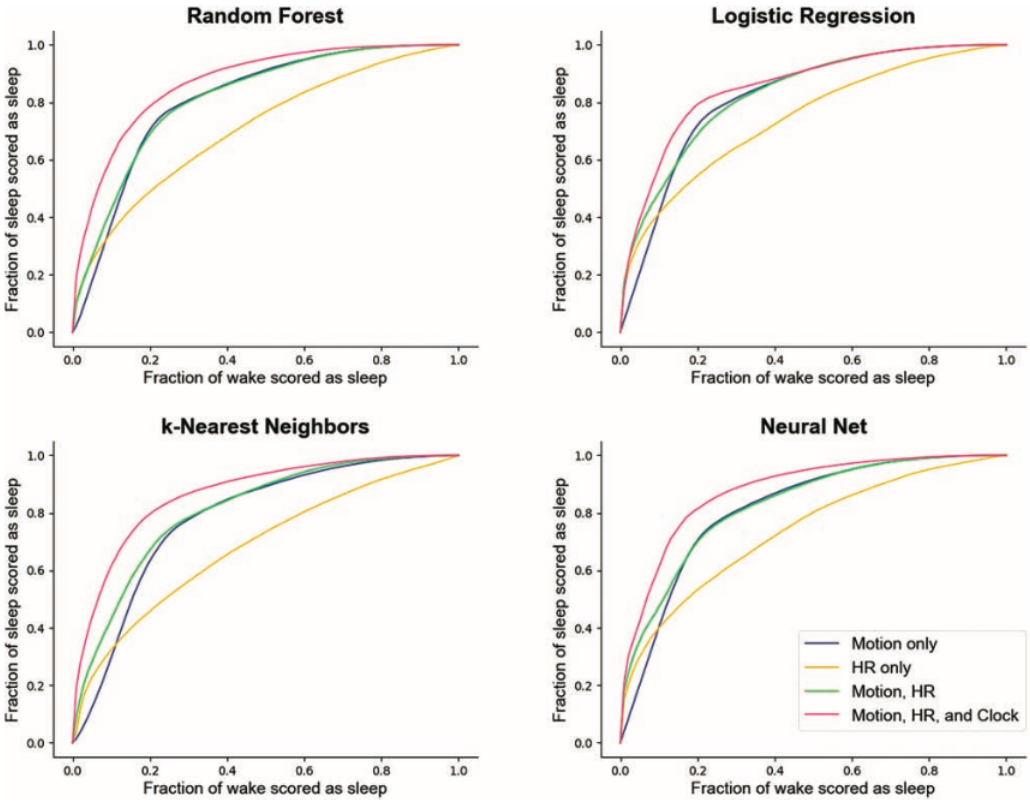


Figure 2.25: Analysis Results

2.9 Integration of the codes

The idea is to integrate this work in the electronic system designed to assess sleep disorders. The overall code is structured in an Object Oriented way, it's more easily to implement an integration of different codes adding classes and instantiates them in the main script.

The goals achieved are: the integration of the preprocessing phase of the Apple Watch raw data with the motion data coming from the Sensortile; in this way data coming from these two different devices are manipulated in order to store them in the same data structure, and the preprocessing steps can be applied to these heterogenous data. This preprocessing step are integrated in the GUI interface. Also the analysis phase is integrated with the GUI interface, it is possible to triggerer the scripts to make these new tasks, preprocessing and analysis of heterogeneous data coming from different commercial devices.

The results of the analysis can be stored in the database updated sending SQL queues from the GUI interface. Also the results plots and logs can be directly seen by the GUI.

Chapter 3

Improvements & Conclusions

3.1 Streaming of Data on a Server & Real Time Processing of Data

The main improvement might be allowing the infrastructure to be *Scalable*.
The idea is to be released from the hospital resource, so that all patients can make their exam at home, using their smart devices.
The general idea is to allow many people to acquire their data, and they want to transmit raw data to the server station. Probably what happen is that different devices are trying to send data at the same time, so is necessary to ensure that the overall infrastructure is able to collect these data.
All data can be stored and analyzed in a 'STREAMING WAY' using a Server, this concept will be applied to overcome the scalability problem.
Big data refers to dynamic, large, and disparate volumes of data that is being created by people, tools, and machines. It refers to data sets that are so massive, so quickly built, and so varied that they defy the traditional analysis methods that you might perform with a relational database. There is no one definition of big data, but there are certain elements that are common across the different definitions, such as velocity, volume, variety, veracity, and value. These are the main 5 Vs of big data. Let's look at each one of them in more detail.

- Velocity: is the speed at which data accumulates. Data is being generated extremely fast, in a process that never stops. The attributes include near- or real-time streaming, local, and cloud-based technologies that can process information quickly.
- Volume is the scale of the data or increase in the amount of data stored. The drivers of volume are the increase in the data sources, higher resolution sensors, and scalable infrastructure.

- Variety is the diversity of the data. Structured data fits neatly in rows and columns in relational databases while unstructured data is not organized in a predefined way, like tweets, blogs, pictures, and videos. Variety also reflects that data comes from different sources like machines, people, and processes, both internal and external to organizations. And the drivers for variety are mobiles, social media, wearable technologies, geo technologies, video, and many more.

- Veracity is the quality and the origin of data and its conformity to facts and accuracy. Attributes include consistency, completeness, integrity, and ambiguity. The drivers include cost and the need for traceability. With large amounts of data available, the debate rages on about the accuracy and authentication of data in the digital age.

- Value refers to our ability and need to turn data into value. The value isn't just profit. It might have medical or social benefits as well as customer, employee, or personal satisfaction. This last V is one of the main reasons why people invest time into big data. They are looking to derive value from it.

The most active open source community projects about Big Data Analytics is *Spark*, and it is advertised as a "lightning-fast unified analytics engine." Spark provides a fast data processing platform that lets you run programs up to 100x faster in memory and 10x faster on disk when compared to Hadoop. Spark also makes it possible to write code quickly, and to easily build parallel apps because it provides over 80 high-level operators. Apache Spark consists of a rich set of SQL queries, machine learning algorithms, and complex analytics, which allows analytics to be performed in a better fashion.

3.2 Conclusions

Wearable smart devices has shown to be one candidate to overcome the problems connected with standard PSG exam to assess sleep disorders.
The SensorTile device from STMicroelectronics performs better than the Apple Watch for what concern the inertial motion tracking, and the Apple Watch acquires the heart rate that has to be integrated with the motion data. The heterogeneous data integration has shown an improvement in the performances of sleep stages classification, this proofs that smart devices may be used as data acquisition system to overcome the invasiveness problem of PSG. In order to further increase the performance, in particular for the NREM/REM discrimination within sleep stages, that can be observed integrating the eye movement during sleeping.
Raw data are integrated and analyzed with AI techniques that has shown really good performances. All the results and the raw data are stored in a database, and a GUI interface allows a connection with to database to import locally the raw data wanted to be analyzed. After the training phase will all data of the database, the Machine Learning algorithms can be run in inference, and the results can be consulted to support the diagnosis directly from the GUI.
This work is supposed to help on the development of an infrastructure that standardize the flow to track sleep disorders in people. The steps to effectively design and develop all the infrastructure are a lot, and requires expertise in different fields, but i hope that this work gives the connection schema between all the elements to perform this alternative method to perform sleep disorders assessment, focusing majorly on the data acquisition system and the data analysis platform.

References

https://www.st.com/en/evaluation-tools/steval-stlkt01v1.html#documentation
https://medium.datadriveninvestor.com/compilation-process-db17c3b58e62
Heterogeneous Data Source Integration and Evolution Mokrane Bouzeghoub , Bernadette Farias Lóscio , 11 Zoubida Kedad , and Assia Soukane
https://towardsdatascience.com/data-preprocessing-concepts-fa946d11c825
https://towardsdatascience.com/dimensionality-reduction-explained-5ae45ae3058e
https://medium.com/codex/feature-scaling-in-machine-learning-e86b360d1c31
Andreas Holzinger, in Encyclopedia of Biomedical Engineering, 2019
https://www.researchgate.net/publication/311488672
https://www.researchgate.net/publication/281746607
https://towardsdatascience.com/machine-learning-classifiers-a5cc4e1b0623
NEUROPSYCHOPHARMACOLOGY REVIEWS, Sleep in Parkinson's disease, Ambra Stefani and Birgit Högl
www.sciencedirect.com
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5428792/
https://iovs.arvojournals.org/article.aspx?articleid=2690287
https://doi.org/10.3389/fnins.2021.616760
SQL and data analysis. Some implications for data analysits and higher education Marin Fotachea, Catalin Strimbeib
Sleep stage prediction with raw acceleration and photoplethysmography heart rate data derived from a consumer wearable device Olivia Walch, Yitong Huang, Daniel Forger and Cathy Goldstein
https://pubmed.ncbi.nlm.nih.gov/23633761/
MCU-Controlling Based Bluetooth Data Transferring System Jia LIU, Guangmin SUN, Dequn ZHAO, Xu YAO, Yihang ZHANG
https://developer.ibm.com/articles/introduction-to-big-data-analysis-with-pyspark/
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3379160/