Universidad Complutense de Madrid

Facultad de Informática

Trabajo de Fin de Grado

Seguimiento automatizado de variables de control en pacientes crónicos

Álvaro Allegue Lorenzo
Ana Carolina Rueda Silva

Dirigido por

Javier Arroyo Gallardo

Departamento de Ingeniería del Software e Inteligencia Artificial

Madrid 2016

Autorización

Los autores de este proyecto autorizan a la Universidad Complutense de Madrid a difundir y utilizar el presente trabajo de investigación, tanto la aplicación como la memoria, únicamente con fines académicos, no comerciales y mencionando expresamente a sus autores. También autorizan a la Biblioteca de la UCM a depositar el trabajo en el Archivo Institucional E-Prints Complutense.

Autores

Álvaro Allegue Lorenzo

Ana Carolina Rueda Silva

Tutor

Javier Arroyo Gallardo

Agradecimientos

Solo al acabar un proyecto uno es realmente consciente de todo el trabajo que ha costado y de las personas que han estado ahí para sacarlo adelante. Por eso queríamos dedicar esta sección a todas aquellas personas que nos han apoyado y nos han dado ánimos para llegar hasta aquí, incluyendo amigos, familiares y profesores. A Alberto y Eva ya que de no ser por ellos este trabajo no existiría.

En especial queremos agradecer a nuestro profesor y tutor del trabajo Javier Arroyo Gallardo por aceptar y hacer más interesante nuestra investigación y guiarnos a lo largo de todo el desarrollo. Por estar ahí cuando le necesitamos y por su paciencia. Pero, sobre todo, por creer en nosotros más de lo que lo hacíamos nosotros mismos.

Por último, queremos agradecer a nuestros amigos que han estado con nosotros en las buenas y en las malas y que nunca dejaron de apoyarnos.

Resumen

Un sueño escaso o de mala calidad puede tener repercusiones negativas en la vida diaria. Si esta situación se prolonga en el tiempo, puede afectar al estado de ánimo e interferir en el trabajo y la vida social de una persona. Son muchos los factores que determinan la calidad del sueño. No es suficiente con dormir, sino que hay que dormir bien para que el sueño sea realmente reparador y se pueda uno levantar descansado y con energía. Muchos aspectos como la actividad física, las comidas y bebidas, los horarios, pueden alterar la calidad del sueño o generar dificultades para conciliarlo por lo que en los últimos años han salido al mercado varios dispositivos y aplicaciones cuantificadoras que ayudan a monitorizar el sueño.

Existen dispositivos que solamente registran el movimiento y la temperatura corporal y aun así generan una gran cantidad de datos. Debido a esto, la información producida puede no ser fácilmente interpretable. En este proyecto usamos un dispositivo de monitorización que ofrece más información (como temperatura y flujo térmico) por lo que queremos mejorar la interpretabilidad de los datos que se generan para ayudar a facilitar el trabajo de los médicos que las utilizan con sus pacientes intentando descubrir trastornos metabólicos. Se aspira a simplificar el trabajo manual que ahora se realiza intentando que sea más automático.

Para ello, en este proyecto se utilizan técnicas de minería de datos en series temporales que permiten hacer un análisis exploratorio de los datos y agrupar la información de forma que los resultados se muestren de una forma resumida y concreta y que pueda ser percibida de un vistazo. Utilizando estas técnicas creamos una aplicación que obtiene series temporales de los datos particionadas en distintos tipos de episodios, centrándonos en episodios de sueño y de actividad física, clasificando los episodios de sueño por su similitud y presentando un resumen de la actividad del paciente a lo largo de toda la monitorización.

Palabras clave

Monitorización del sueño, movimiento, minería de datos, series temporales, dynamic time warping, clustering, Python.

Abstract

Sparce or poor quality sleep can have a negative impact on daily life. If this situation continues over time, it can affect the mood and interfere with work and social life. There are many factors that determine the quality of sleep. It is not sufficient with sleeping every day, but must sleep well so that sleep is a really restful sleep and one can lift rested and energized. Many aspects such as physical activity, meals and drinks, schedules, can disrupt sleep quality or cause difficulty falling asleep so in recent years have come on the market several bracelets or armbands and quantifying applications that help monitor the sleep.

There are devices that only record the movement and body temperature and yet generate a large amount of data. Because of this, the information produced can not be easily interpretable. In this project we use a monitoring device that provides more information (such as temperature and heat flow) so we want to improve the interpretability of the data generated to help ease the work of doctors who use it with their patients trying to discover metabolic disorders. We aim to simplify the manual work that is now performed trying to be more automated.

For this purpose, in this project time series data mining techniques are used that allow an exploratory analysis of the data and group information so the results are displayed in a summarized and concrete way and allow to observe them at a glance. Using these techniques, we have created an application that obtains time series data partitioned into different types of episodes, focusing on episodes of sleep and physical activity, classifying sleep episodes by their similarity and a doing a summary of patient activity throughout of all monitoring.

Keywords

Sleep monitoring, movement, data mining, time series, dynamic time warping, clustering, Python.

Índice de contenidos

Autorización I

Agradecimientos III

Resumen V

Abstract VII

Índice de figuras XIII

I Introducción 1

1 Introducción 3

1.1 Objetivo 3

1.2 Problemas asociados y metodología de trabajo seguida 4

II Estado del arte 7

2 Dispositivos de monitorización en la actualidad 9

2.1 Dispositivos basados en señales de la actividad cerebral 9

2.1.1 Zeo 9

2.2 Dispositivos basados en señales autonómicas 10

2.2.1 M1 Sleepimage 10

2.3 Dispositivos basados en movimiento 11

2.3.1 Fitbit 11

2.4 Otros 12

2.4.1 Apple Watch 12

2.4.2 Camiseta inteligente (Universidad Carlos III de Madrid) 13

2.4.3 Aplicaciones móviles 13

2.4.4 LifeBed de Hoana 13

2.4.5 Monitores de sueño en cama 13

3 Dispositivo utilizado 15

3.1 Especificaciones 15

3.2 Mediciones 16

3.3 Limitaciones 17

3.4 Software proporcionado por el fabricante 17

3.5 Finalidad o utilidad médica 19

III Métodos de minería de datos 21

4 Minería de datos de series temporales 23

4.1 Introducción 23
4.2 Preprocesado 25
4.2.1 Tratamiento de los valores nulos o del ruido 25
4.2.2 Normalización de los datos 26
4.3 Medidas de similitud de series temporales 28
4.3.1 Distancia en series temporales 28
4.4 Clustering 35
4.4.1 Las metas del clustering 35
4.4.2 Posibles aplicaciones 36
4.4.3 Requisitos 36
4.4.4 Problemas 36
4.4.5 Algoritmos de clustering 36
4.5 Predicción 43
4.5.1 Alisado exponencial 43
4.6 Resumen 45
4.6.1 Representación 45
4.6.2 Visualización 47

IV Aplicación 49

5 Descripción de la aplicación 51

5.1 Porqué usar Python y no R 51
5.1.1 Análisis de datos 51
5.1.2 ¿Qué convierte a Python en el más extendido? 51
5.1.3 ¿Por qué usar R para análisis de datos? 52
5.1.4 Limitaciones 52
5.1.5 Conclusiones 52
5.2 Objetivo de la aplicación 53
5.3 Descripción de la funcionalidad 53
5.3.1 Pestaña 1: Visualización de episodios de sueño 55
5.3.2 Pestaña 2: Relaciones entre temperaturas durante el sueño 59
5.3.3 Pestañas 3 a 5: Clustering de sueños 62
5.3.4 Pestaña 6: Consumos 67
5.5 Manejo de los datos 69
5.6 Descripción del código y arquitectura 69
5.7 Tecnologías utilizadas 71

V Conclusiones y trabajo futuro 73

6 Conclusiones 75

7 Conclusions 76

8 Trabajo futuro 77

VI Organización del proyecto 79

9 Organización del proyecto 81

9.1 Organización del proyecto 81

9.2 Contribución al proyecto 82

9.2.1 Álvaro Allegue Lorenzo 82

9.2.2 Ana Carolina Rueda Silva 84

VII Anexos 89

A. Ética y privacidad de los datos 91

B. Manual de instalación 93

Bibliografía 97

Índice de figuras

Figura 2.1: Monitor de sueño de Zeo. 10

Figura 2.2: Monitor de sueño M1 de SleepImage. 11

Figura 2.3: Monitor de actividad de Fitbit. 12

Figura 2.4: Apple Watch. 12

Figura 3.1: Brazalete Bodymedia SenseWear. 15

Figura 3.2: Sensores del brazalete Bodymedia SenseWear. 16

Figura 3.3: Captura de SenseWear Professional. 17

Figura 3.4: Informe en la aplicación SenseWear Professional. 18

Figura 3.5: Informe generado por aplicación SenseWear Professional. 19

Figura 4.1: Ejemplo de series normalizadas. 26

Figura 4.2: Clasificación de los distintos tipos de distancias entre series temporales numéricas. 28

Figura 4.3: Comparación de dos series temporales mediante distancia euclídea. 28

Figura 4.4: Distancias Minkowski para $p = 1$ (Manhattan), $p = 2$ (Euclídea) y $p = \infty$ (Chebyshev). 29

Figura 4.5: Secuencias que coinciden aproximadamente y transformación de escala 30

Figura 4.6: Aplicación de la distancia euclídea y DTW a dos series. 30

Figura 4.7: Ejemplo de alineación con DTW 31

Figura 4.8: Ejemplo de condición de contorno. 32

Figura 4.9: Ejemplo de restricción de la pendiente. 33

Figura 4.10: Ejemplo de ventana de ajuste y continuidad. 33

Figura 4.11: Ejemplo de clustering. 35

Figura 4.12: Ejemplo de dendrograma. 37

Figura 4.13: Ejemplo de agrupamiento con SLINK en datos Gaussianos. 38

Figura 4.14: Ejemplo de K-means. 40

Figura 4.15: Esquemas de Voronoi resultantes de ejecutar K-means 40

Figura 4.16: Ejemplo de método Elbow. 41

Figura 4.17: EM usado en datos de una distribución normal 41

Figura 4.18: Ejemplo de agrupamiento basado en densidad con DBSCAN. 42
Figura 4.19: Clasificación de los tipos de representación de series temporales. 45
Figura 4.20: Visualización de la técnica de reducción de dimensionalidad PAA 46
Figura 4.21: Visualización de la técnica de reducción de dimensionalidad SAX. 47
Figura 4.22: Ejemplo de visualización en espiral de una serie periódica. 48
Figura 4.23: Captura de pantalla de VizTree. 48
Figura 5.1: Selección de fichero a cargar 53
Figura 5.2: Nombres de episodios 54
Figura 5.3: Pestaña del intérprete de sueños de la aplicación 55
Figura 5.4 Selección de episodios de sueño 56
Figura 5.5 Tipos de episodios en la gráfica de barras 56
Figura 5.6: Gráfica de postura 57
Figura 5.7 Curvas de movimiento y consumo 57
Figura 5.8 Filtro y selector de episodios 59
Figura 5.9 Información del episodio visualizado 60
Figura 5.10: Ejemplo de interpretación de coeficiente de correlación. 60
Figura 5.11: Captura de gráficas de flujo temporal y temperatura 61
Figura 5.12: Diagrama de dispersión 61
Figura 5.13 Pestaña de clustering 62
Figura 5.14: Selector de tipo de clustering 63
Figura 5.15: Filtro de episodios de sueño. 63
Figura 5.16: Matriz de distancias 64
Figura 5.17: Captura de dendrograma 64
Figura 5.18: Ejemplo de grupos similares en dendrograma 65
Figura 5.19: Comparación de dos episodios de sueño 66
Figura 5.20: Pestaña consumos de la aplicación 67
Figura 5.21: Gráfico de barras 67
Figura 5.22: Gráficas de consumo correspondientes a un día 68

Parte I

Introducción

1 Introducción

Numerosos estudios demuestran que la actividad física repercute directamente en la duración y calidad del sueño.

Según el estudio [1], las personas duermen significativamente mejor y se sienten más activas durante el día si han realizado un mínimo de unos 150 minutos de actividad física a la semana. Se usó una muestra representativa de más de 3000 hombres y mujeres de edades comprendidas entre los 18 y los 85 años, teniendo en cuenta su índice de masa corporal, estado de salud, y además si son o no fumadores y/o sufren depresiones, que dio como resultado que al realizar 150 minutos de ejercicio moderado a intenso a la semana se obtiene una mejora del 65% en la calidad del sueño.

Las personas que realizan ejercicio afirmaron sentir menos sueño durante el día en comparación con los que realizan menor actividad física [2]. También se obtuvieron resultados similares para el síndrome de piernas inquietas (68% de menor probabilidad) y para la dificultad de concentrarse cuando se está cansado (45% menos probable).

El dormir mal puede afectar negativamente a la productividad del individuo en el trabajo o en el caso de un estudiante a su atención en clase y por lo tanto su rendimiento académico. En Estados Unidos, sobre el 35-40% de la población adulta padece problemas para conciliar el sueño o tiene problemas de somnolencia diurna [1]. El sueño insuficiente y la alteración del mismo se asocia con numerosos resultados negativos para la salud en adultos, incluyendo la depresión clínica y otras condiciones perjudiciales como las enfermedades cardiovasculares, la diabetes tipo 2, la obesidad y la mortalidad [3].

Estos hechos son la razón de que se hayan desarrollado varias soluciones de monitorización del sueño. Gracias a estos dispositivos se puede detectar las posibles alteraciones que puedan estar ocasionando un sueño de mala calidad. La dificultad está en analizar los datos que generan, ya que por lo general su interpretación no es sencilla y muchas veces requiere de la ayuda de un especialista.

Como resultado de estas reflexiones, la motivación principal de este proyecto es la de ayudar al especialista a trabajar mejor con los datos para diagnosticar a sus pacientes.

1.1 Objetivo

El objetivo principal de este trabajo consiste en crear una aplicación que trabaje con una serie de datos generados por un dispositivo de monitorización y que permita facilitar el trabajo de los expertos. Para ello contamos con la colaboración del doctor Alberto Ordóñez Pérez y de la enfermera Eva Pacho, de Ibermutuamur, que, además de proporcionarnos las herramientas necesarias nos ayudan a ver lo que podría ser útil analizar en la aplicación.

El dispositivo es un brazalete con distintos sensores, empleado generalmente por deportistas en centros de alto rendimiento y de un alto coste económico, por lo que hasta ahora no se había utilizado con fines médicos.

Dada la posibilidad de que el usuario medio que vaya a utilizar la aplicación carezca de conocimientos informáticos avanzados, es necesario incluir en los objetivos una interfaz simple y clara. A la hora de diseñar el programa también cobra importancia el que sea capaz de realizar la misma labor ante posibles cambios de configuración del dispositivo, ya que puede ser necesario modificar los valores iniciales para adecuarlos a cada paciente. Así pues, lo primero que tenemos que

hacer es conseguir leer los datos con los que vamos a trabajar y que éstos no generen problemas. A partir de este punto dividimos la aplicación en cuatro fases:

1. **Pre-procesar** los datos para extraer sólo la información válida. No utilizamos todos los parámetros registrados por el brazalete y descartamos aquellos datos inválidos.

2. **Procesar** los datos para filtrar la información importante que presentar al usuario. Dependiendo del apartado de la aplicación particionamos los datos en días, episodios de sueño o de actividad física. También separamos los episodios de sueño entre diurnos y nocturnos.

3. **Analizar** los datos mediante técnicas de análisis de datos, clasificando la información por clustering de series temporales. Esto nos permite agrupar episodios por su similitud con la finalidad de encontrar patrones y anomalías entre los episodios de sueño.

4. **Visualizar** los resultados. De forma lo más sencilla posible sin requerir una gran interacción y siguiendo un orden que permita ir descubriendo la información de forma racional.

Por lo tanto, se trata de realizar un análisis de los datos obtenidos por el dispositivo que permita al especialista detectar anomalías en los episodios de sueño del paciente. Estas anomalías podrían sugerir posibles trastornos en el sueño e indicar el estilo de vida que lleva en cuanto a actividad física y consumo energético se refiere. Para ello partimos de técnicas de exploración como el clustering para intentar obtener información de utilidad a partir de los datos disponibles. Posteriormente diseñamos una aplicación que sea capaz de representar los datos de una forma sencilla que requiera poca interacción por parte del usuario y enfocada la visualización de episodios de sueño y de actividad física, con agrupamiento de los episodios de sueño por similitud y un resumen del consumo calórico del paciente.

1.2 Problemas asociados y metodología de trabajo seguida

El primer inconveniente encontrado fue confirmar si era posible extraer los datos brutos registrados por el brazalete en algún formato que se pudiera leer, como *csv*. En un principio únicamente conocíamos el software proporcionado por el fabricante y solamente con él nos resultaría imposible realizar ninguna aplicación o estudio de minería de datos, ya que trata directamente con los datos y los muestra mediante gráficas, lo que haría imposible manipularlos. Finalmente, tras poder probar el programa por nuestra cuenta nos cercioramos de que existía la opción de exportar los datos en formato csv y xls.

Otro problema relacionado con los datos es su fiabilidad. El brazalete no siempre recoge datos válidos, ya que tanto al colocarlo como al quitarlo es posible que registre información errónea que es necesaria descartar para no alterar los resultados obtenidos. Cabe la posibilidad de que el paciente se retire el brazalete durante la monitorización, por ir a la ducha, a la piscina u otras actividades incompatibles con su uso. Por lo tanto, se obtendrán saltos temporales que no vamos a descartar, sino que representaremos de igual forma ya que se supone que el propio paciente debe informar de estos acontecimientos, que quedarán registrados en el diario.

Al ser una colaboración entre dos mundos tan diferentes como la medicina y la informática, nos llevó varias reuniones establecer unos objetivos y requisitos iniciales para el proyecto. Empezamos documentándonos sobre temas médicos que pudieran estar relacionados con los brazaletes siguiendo las directrices del doctor, tales como trastornos del sueño, trastornos metabólicos, fases del sueño, entre otros y sobre la funcionalidad del brazalete y la forma de registrar los datos,

consultando las dudas que nos iban surgiendo.

El profesor Javier Arroyo nos proporcionaba nuevas ideas y evaluaba el trabajo realizado tras cada reunión. Posteriormente nos reuníamos en Ibermutuamur donde presentábamos los avances y preguntábamos las dudas que nos iban surgiendo. De esta forma fuimos desarrollando varios programas independientes que después agrupamos en la aplicación final. Por lo tanto, durante todo el desarrollo hubo que afrontar cambios en los requisitos, lo que implicaba descartar algunos avances y adaptar la estructura del código para implementar las nuevas ideas, por ello la estructura debía ser lo más flexible y abierta posible a futuros cambios.

Parte II

Estado del arte

2 Dispositivos de monitorización en la actualidad

Aunque el estándar para evaluar el sueño sigue siendo el laboratorio de sueño (polisomnografía) existe un interés creciente por los dispositivos de vigilancia portátiles que proporcionan la oportunidad de evaluar el sueño en entornos del mundo real, como el hogar o el trabajo [4]. Los dispositivos portátiles permiten múltiples mediciones, la evaluación de patrones temporales y la auto-experimentación.

Desde el punto de vista clínico y de investigación, la capacidad de obtener datos de sueño-vigilia longitudinales pueden mejorar el fenotipo de la enfermedad, y las decisiones de tratamiento y la optimización de la salud individualizadas.

Desde el punto de vista del bienestar, los dispositivos disponibles en el mercado permiten al individuo realizar un seguimiento de su propio sueño con el objetivo de encontrar patrones y correlaciones modificables en su estilo de vida tales como el ejercicio, la dieta y medicamentos para el sueño.

El laboratorio de polisomnografía (PSG) ha demostrado ser muy útil para el diagnóstico y tratamiento de la apnea obstructiva del sueño (AOS), aunque los trastornos menos comunes también son fácilmente identificables incluyendo la narcolepsia, el sueño de movimientos oculares rápidos (REM), parasomnias no-REM y el síndrome de piernas inquietas entre otros.

A pesar de la clara utilidad del PSG en medicina, tiene bastantes inconvenientes que hacen imposible realizar la prueba a cualquier paciente. Algunos de estos inconvenientes son el coste y su dificultad de uso y aparatosidad, ya que se necesita una habitación con máquinas conectadas al paciente mediante una gran cantidad de cables, además de especialistas que preparen la prueba y permanezcan junto al paciente durante toda la medición además de la posterior interpretación de la información. Estos inconvenientes han motivado el desarrollo de dispositivos portátiles capaces de evaluar el sueño en el hogar y fuera de él.

En un principio estos dispositivos estaban orientados al diagnóstico de apneas, pero con el tiempo fueron desarrollándose nuevos para pacientes con insomnio y trastornos del sueño más habituales para cuya detección no era necesaria una prueba de polisomnografía. Las primeras pulseras de actigrafía[1] se colocaban en la muñeca y registraba movimientos del paciente durante el sueño. Era útil para la documentación de patrones de sueño

A continuación, se lista algunos de los dispositivos que se han utilizado para distintos tipos de mediciones.

2.1 Dispositivos basados en señales de la actividad cerebral

2.1.1 Zeo

El dispositivo de Zeo es pequeño, con un diseño muy parecido al de un pulsómetro con la única diferencia de que se ha de colocar en la cabeza a modo de diadema [5]. Este dispositivo consta de una cinta elástica con sensores de tela que se colocan en la frente. Estos sensores detectan una combinación de electroencefalograma (EEG), electromiograma del músculo frontalis (EMG), y las señales del electrooculograma (EOG). Transmite las señales para su análisis de forma inalámbrica,

[1] Permite dar una idea indirecta de la estructura del sueño a través de la actividad física y movimientos del paciente.

ya sea a una estación receptora de un despertador o a un iPhone. La principal ventaja de Zeo es la capacidad de controlar el sueño en el tiempo con cierta facilidad. De hecho, el dispositivo puede ser usado diariamente durante varios meses antes de que los paneles sensores necesiten ser reemplazados.

Figura 2.1: Monitor de sueño de Zeo.

Un modelo patentado de red neural utiliza los flujos de datos para hacer clasificaciones de vigilia y las distintas fases del sueño en intervalos de 30 segundos. El sueño profundo, que cuenta con categoría propia, se corresponde con el sueño de onda lenta o fase 3 (N3). Las fases 1 y 2 (N1 y N2) son clasificadas en una sola categoría generando así un problema ya que la etapa N2 representa la mayor parte del tiempo de sueño en un individuo normal y no hay forma de distinguir estas dos etapas al utilizar este dispositivo.

Se desconoce el grado en el que el algoritmo clasificador conserva la precisión con pacientes que sufren de trastornos del sueño, neurológicos o psiquiátricos debido a la posibilidad de que ciertos medicamentos alteren su precisión. Además, también son desconocidos los efectos de la cafeína, teína, alcohol y el tabaco (todos los cuales se sabe que afectan a la fisiología del sueño). Zeo permite a los investigadores realizar un post-procesamiento de las señales registradas, aunque no se especifica cómo.

2.2 Dispositivos basados en señales autonómicas

2.2.1 M1 Sleepimage

A diferencia de la primera generación de dispositivos que medían el movimiento y, a veces, la frecuencia cardíaca, los nuevos instrumentos clínicos de consumo, como el M1 SleepImage, pueden medir el ritmo cardiaco (ECG), el volumen de respiración e incluso el de los ronquidos a través de la vibración de los tejidos. También pueden mantener control sobre el movimiento del cuerpo o la posición. Esta relación entre ritmo cardiaco y respiratorio se asigna a las fases del sueño y trastornos respiratorios que hasta ahora sólo una polisomnografía podía medir.

Figura 2.2: Monitor de sueño M1 de SleepImage.

El dispositivo M1 de SleepImage es un disco pequeño, ovalado, que se coloca durante la noche en el pecho, junto al corazón. Incorpora también un electrodo que se fija más abajo, en las costillas [6]. Pasada la noche, se retira y se descargan los datos, mostrando un desglose objetivo sobre el sueño:

- Etapas de sueño estables frente a inestables,
- sueño REM,
- recuento de ronquidos,
- posición e interrupciones del sueño.

Ofrece más datos biométricos que pueden ser útiles para distinguir patrones cardiopulmonares de la apnea obstructiva del sueño y otros trastornos respiratorios relacionados con el sueño.

Una limitación de este dispositivo es que ciertos pacientes pueden no ser susceptibles de análisis de ECG[2], incluyendo los que tienen ciertos tipos de arritmias y, potencialmente, los pacientes con disfunción autonómica. Además, la actigrafía torácica como la que proporciona este dispositivo no tiene tantos datos de apoyo para estimar el sueño y la vigilia, en comparación con la actigrafía de muñeca tradicional.

2.3 Dispositivos basados en movimiento

2.3.1 Fitbit

El monitor de Fitbit es un pequeño dispositivo que se puede llevar en la muñeca, sujeto en la ropa, o llevarse en un bolsillo. Incluye un podómetro y un altímetro (para contar pasos o altura escalada). Tiene además una función de conteo de calorías (extrapolada a partir de la estimación de pasos dados), detecta el movimiento por actigrafía, y además incorpora un reloj.

[2] Electrocardiograma

Figura 2.3: Monitor de actividad de Fitbit.

El análisis del movimiento proporciona métricas estándar relacionadas con el sueño como una distinción entre el sueño y la vigilia, el tiempo total de sueño, la latencia del sueño y un "índice de despertares", basado en episodios de movimiento durante el presunto tiempo de sueño. No hay validaciones publicadas de la exactitud de las mediciones de sueño-vigilia de Fitbit en comparación con el PSG o para dispositivos de visualización estándar de actigrafía.

2.4 Otros

2.4.1 Apple Watch

Aunque no es una de sus funciones más destacadas, el reloj cuenta con todos los sensores necesarios para poder efectuar una medición precisa del sueño (movimiento y pulsaciones), y ahora ya hay aplicaciones que pueden medir de forma autónoma, sin necesidad de estar conectadas al iPhone, que cumplen este cometido.

Figura 2.4: Apple Watch.

Más recientemente se ha lanzado una aplicación para monitorizar el sueño que forma parte de una investigación sanitaria integrada con HealthKit [7]. Esta aplicación no sólo mide los hábitos de sueño, sino que además los combina con otros datos (ritmo cardíaco, función Night Shift de iOS) para estudiar el impacto que tiene una noche de buen o mal sueño en la vida diaria.

2.4.2 Camiseta inteligente (Universidad Carlos III de Madrid)

El uso de esta prenda permite registrar de modo no intrusivo una serie de parámetros fisiológicos de un paciente. "La información captada a través de una camiseta inteligente con tecnología e-textil se envía de forma inalámbrica a un sistema gestor, el cual muestra en tiempo real la localización y constantes vitales de los pacientes", explican los investigadores de la UC3M [8] [9]. Esta camiseta es lavable e integra unos electrodos que detectan el potencial bioeléctrico a partir del cual se obtiene el electrocardiograma. Además, cuenta con un dispositivo extraíble que incluye un termómetro y un acelerómetro, con los cuales se obtiene la temperatura del paciente, su posición relativa (en reposo, de pie, etc.) y su índice de actividad física.

2.4.3 Aplicaciones móviles

Las aplicaciones móviles solo sirven para hacer un seguimiento del sueño basándose en los movimientos que se hacen al dormir [10]. Funcionan colocando el móvil bajo la almohada para que el sistema comience a registrar con gran precisión todos los movimientos. Por otro lado, estas aplicaciones cuentan con una opción de despertador en función de los ritmos circadianos que puede resultar útil. Sin embargo, no es posible obtener los datos brutos que genera para poder hacer un análisis sin contar con la aplicación. Algunas de las aplicaciones más conocidas son Sleep Cycle[3], Sleepbot[4], Pillow[5] y Sleep time[6].

2.4.4 LifeBed de Hoana

Es una cama utilizada en entornos clínicos que muestra y registra la tasa de respiración y el ritmo cardiaco, también tiene una función que alerta a los cuidadores cuando un paciente se levanta.

2.4.5 Monitores de sueño en cama

En este apartado se engloban los monitores de sueño en cama. Por lo general, se dividen en dos tipos:

- Dispositivos que se colocan **debajo del colchón** y suelen ser dispositivos piezoeléctricos que miden la respiración, el ritmo cardíaco, ronquidos, tos y el movimiento.

- Dispositivos con forma de cojín fino que se colocan **encima del colchón**. Estos dispositivos cuentan con una almohadilla sensible a la presión que mide la frecuencia cardíaca, la frecuencia respiratoria, el ronquido y el movimiento corporal.

[3] https://www.sleepcycle.com/
[4] https://mysleepbot.com/
[5] https://neybox.com/pillow/
[6] http://www.azumio.com/s/sleeptime/index.html

3 Dispositivo utilizado

El dispositivo que vamos a utilizar para este trabajo es el brazalete Bodymedia SenseWear [11].

Figura 3.1: Brazalete Bodymedia SenseWear.

El monitor metabólico Bodymedia SenseWear Armband monitoriza, mide y cuantifica la actividad física diaria, incluyendo horas de sueño, tiempo de descanso, etc. Es portátil y auto-calibrable evitando la necesidad de acudir a personal muy entrenado para calibrar los calorímetros clásicos. Es un equipo de análisis metabólico multi-sensorial que se coloca sobre uno de los brazos durante una o dos semanas y realiza el cálculo del gasto energético y la cuantificación de la actividad física metabólica [12].

Para ello, registra las señales fisiológicas provenientes de cinco sensores en intervalos de un minuto:

- Acelerómetro en los tres ejes
- Temperatura corporal
- Disipación térmica
- Dos medidores de respuesta galvánica de la piel (impedancia de la piel/grado de humedad)

El brazalete permite recolectar y analizar información sobre los pacientes en cualquier entorno permitiéndoles desarrollar sus actividades diarias sin interferencias.

Esta es la principal diferencia con los dispositivos mencionados anteriormente pues se considera un dispositivo especializado y no dirigido al consumo. Se usa incluso con deportistas en centros de alto rendimiento ya que la información que proporciona sobre los periodos de sueño y vigilia es más completa y puede ser más relevante en el momento de analizarla medicamente.

3.1 Especificaciones

- Peso: 45 gr.
- Dimensiones: 55 x 62 x 13 mm
- Batería interna recargable desde el ordenador (puerto USB) o con cargador USB a red 220 V que dura aproximadamente 5-7 días.
- Capacidad de la memoria: 28 días de uso continuo
- Material dispositivo: ABS, Policarbonato, poliuretano termoplástico, acero inoxidable. hipoalergénico de grado 304.

- Material tira: Nylon, poliéster, policarbonato, lycra.
- Instrumento medical de Clase IIa CE/93/42 Tipo BF CE0086

3.2 Mediciones

El brazalete SenseWear incluye varios sensores adicionales que pueden identificar los cambios fisiológicos del ejercicio que no están asociados con el movimiento. Estos incluyen intercambio de calor, temperatura de la piel y sudor, permitiendo a los algoritmos identificar con precisión tanto el gasto energético como los estados de sueño y vigilia independientemente del movimiento del paciente. Con todos estos sensores el brazalete puede determinar el tipo de ejercicio y la intensidad de la actividad física realizada. Esto es importante porque cada persona quema calorías de manera diferente ya que cada una tiene su metabolismo basal [13], siendo éste el valor mínimo de energía necesaria para subsistir.

Las mediciones que puede hacer el brazalete, haciendo uso de sus 5 sensores, son:

- **Movimiento:** El brazalete contiene un acelerómetro, un dispositivo que mide el movimiento. Se usa para medir el movimiento desde múltiples ejes y perspectivas, permitiendo así definir mejor la actividad realizada.

- **Pasos:** El brazalete cuenta los pasos utilizando el acelerómetro para medir los distintos patrones creados por caminar y / o correr.

- **Respuesta galvánica de la piel:** Mide la conductividad eléctrica de la piel, que cambia en respuesta al sudor y estímulos emocionales. Sirve para medir mejor el nivel de actividad.

- **Temperatura de la piel:** Un termómetro electrónico muy sensible mide la temperatura de la piel.

- **Flujo de calor:** Mide la cantidad de disipación de calor del cuerpo producido por los músculos al moverse.

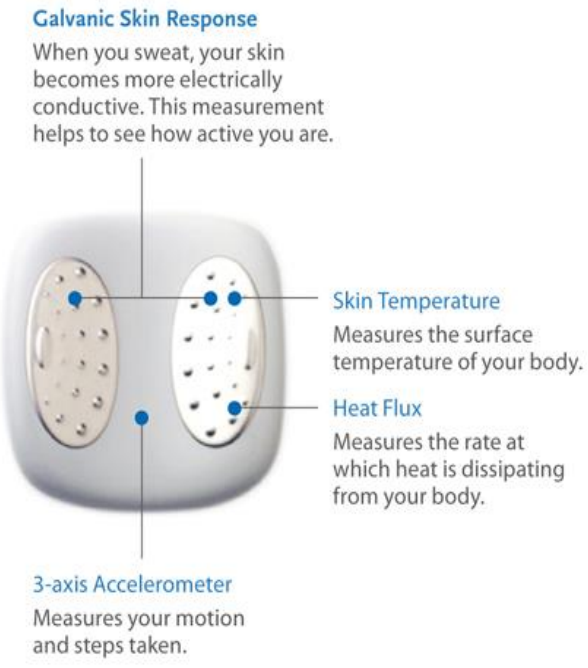


Figura 3.2: Sensores del brazalete Bodymedia SenseWear.

3.3 Limitaciones

El brazalete no es sumergible ni puede entrar en contacto con el agua ni otros líquidos por lo que el paciente debe cerciorarse de retirarlo antes de las duchas y otras actividades que pueda mojar el dispositivo como piscina, sauna etc. Tampoco se debe colocar el brazalete en zonas de la piel donde se haya aplicado alguna crema ya que puede degradar el plástico de la carcasa.

Sólo puede almacenar los datos de un paciente al mismo tiempo durante un máximo de 28 días. Cada vez que se quiera colocar el brazalete en un nuevo paciente hay que volcar los datos del paciente anterior en el ordenador mediante el programa incluido.

3.4 Software proporcionado por el fabricante

La visualización de los datos se realiza, tras un volcado de los mismos a un ordenador a través del puerto USB del brazalete, en una única ventana mediante gráficas temporales especificando periodos de tiempo o eventos concretos (uno o varios días, unas horas...) y pudiendo filtrar los datos con los parámetros obtenidos por los sensores del brazalete.

Se dispone de 24 parámetros en total entre los que destacan la fecha de cada registro, los tres acelerómetros (transversal, anterógrado y longitudinal), temperatura de la piel, temperatura cerca del cuerpo, contador de pasos, acostado (si/no), sueño (si/no), actividad física (si/no), gasto energético, MET[7], velocidad, distancia, clasificación de actividad y sueño, flujo térmico y anotaciones por cada registro.

Algunos de los parámetros vienen dados como la media de la medida durante el minuto que registra o como los picos máximos de esa medida, lo cual resulta útil para según qué mediciones. Por ejemplo, para la detección de movimiento durante el sueño, es mejor observar los picos en un minuto que la media, ya que los movimientos serán rápidos y cortos, lo que dificultaría en gran medida su apreciación observando sólo la media.

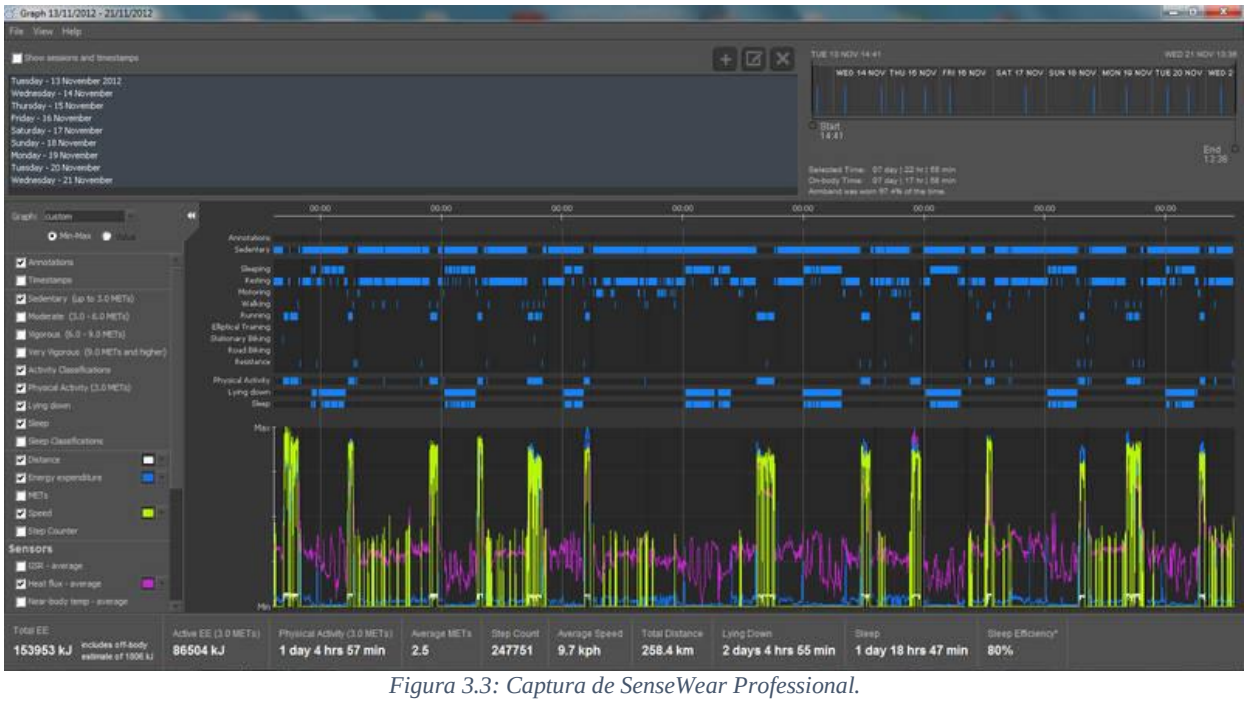


Figura 3.3: Captura de SenseWear Professional.

[7] Unidad de medida del índice metabólico definida como la cantidad de calor emitido por una persona en posición sedente por metro cuadrado de piel.

Al finalizar el periodo de monitorización, se puede obtener un informe personalizado con los siguientes datos:

- Gasto diario total (Kcal consumidas)
- Duración, cuantificación y clasificación de la actividad física
- METS equivalente metabólico (Kcal / HR / Kg)
- Gasto energético activo durante la actividad
- Número de pasos efectuados
- Posición corporal
- Tiempo en pie, sentado y tumbado
- Eficiencia y duración del sueño

Se puede ver un ejemplo del informe personalizado en la Figura 3.4. Este informe es escaso en cuanto a la información que ofrece ya que solo permite agregar o quitar días para ver una acumulación de los datos.

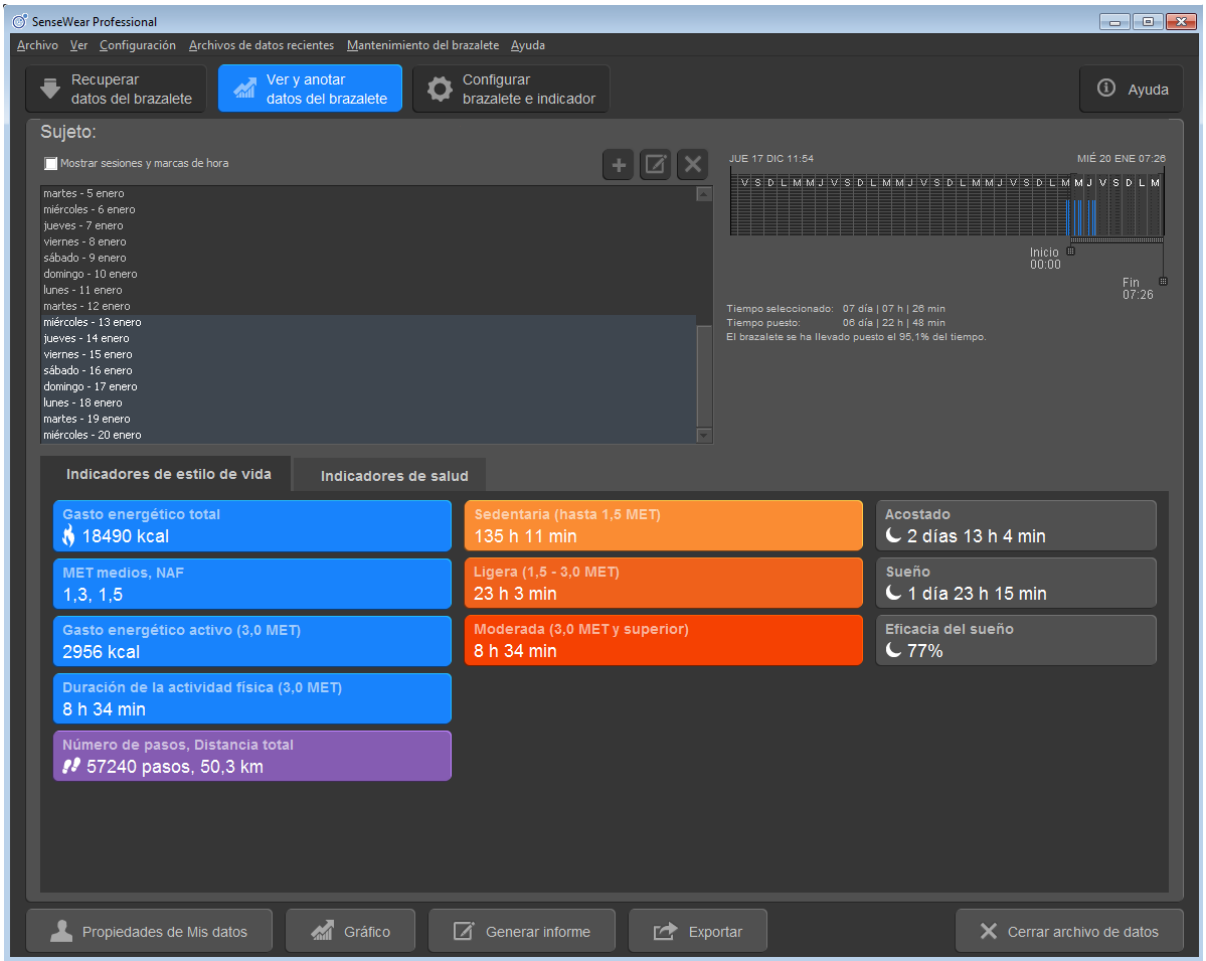


Figura 3.4: Informe en la aplicación SenseWear Professional.

También permite exportar un informe más detallado y visualizar gráficamente los datos. Este informe se genera en formato PDF con representaciones gráficas sobre el estilo de vida del paciente como se puede ver en la Figura 3.5. De todas formas, este informe está muy limitado dado que es estático y solo muestra la suma de los datos recogidos sin distinción del tipo de actividad o tipo de sueño.

Los datos brutos de todas las señales registradas y la información del paciente se pueden exportar en formato *csv* y *xls*, lo que nos permite visualizar los datos con programas como Excel o Matlab, entre otros y poder manipularlos para realizar minería de datos e investigar en detalle sin depender de este programa.

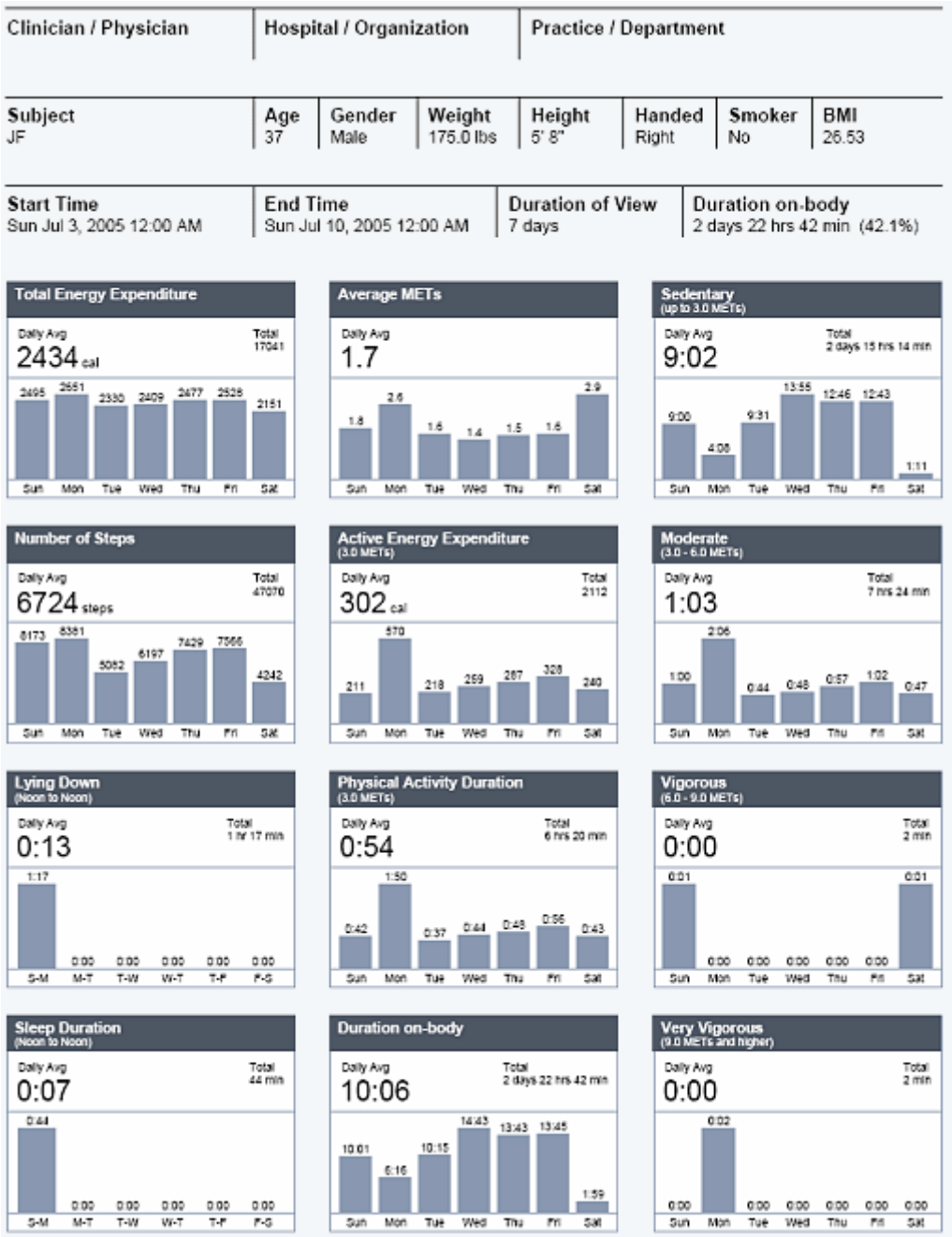


Figura 3.5: Informe generado por aplicación SenseWear Professional.

3.5 Finalidad o utilidad médica

En el departamento de Ibermutuamur colaborador, tratan generalmente con pacientes que han sufrido accidentes laborales y están en periodo de recuperación. También tratan con obesos, deportistas y pacientes con trastornos del sueño.

La finalidad de colocar un brazalete a cada paciente es la de realizar un seguimiento a lo largo de entre una y dos semanas de la actividad física y de los periodos de sueño del paciente para poder aplicarle dietas personalizadas y observar la duración, calidad y patrones de sueño.

El seguimiento se realiza conjuntamente con el paciente, al que se le hacen preguntas sobre el día a día mientras lleva colocado el brazalete y se van anotando en un diario para posteriormente contrastar con la información proporcionada por el dispositivo. Esto resulta muy útil para, por ejemplo, comprobar si la actividad física realizada por el paciente es suficientemente efectiva o para corroborar si se ejercita tanto como indica en el diario.

Transcurrido el tiempo de colocación del brazalete, el paciente lo devuelve y los médicos vuelcan los datos en un ordenador a través del programa SenseWear Professional. Consultan el informe generado por el programa donde suele destacar el consumo basal del paciente, el consumo diario, la cantidad e intensidad de actividad física realizada y la eficacia de sueño. Con estos datos y el diario, dependiendo del diagnóstico informan al paciente de los cambios que debe realizar en su estilo de vida para mejorar su situación. Estos cambios engloban cambiar la dieta, la frecuencia, intensidad y hora del ejercicio físico y la rutina de sueño entre otras cosas.

También utilizan el programa para observar con más detalle eventos puntuales que se hayan podido producir durante el tiempo en que el paciente llevaba el brazalete. Por ejemplo, pueden ver cómo se desarrollan los episodios de sueño, si alcanza siempre el sueño profundo, si se despierta frecuentemente, si tarda mucho en dormir desde que se acuesta, si se mueve mucho, etc. Puede servir como un primer diagnóstico ante posibles apneas del sueño, antes de realizar una polisomnografía, ya que esta prueba es cara y no siempre necesaria.

Por lo tanto, aunque se trate de un dispositivo originalmente ideado para deportistas y dietética en general, puede resultar de gran utilidad médica dada su facilidad de uso, precisión y coste respecto a otras pruebas más precisas y caras como la polisomnografía.

Parte III

Métodos de minería de datos

4 Minería de datos de series temporales

A continuación, se da una introducción sobre minería de datos y más especialmente la minería de datos de series temporales, en la que nos vamos a centrar dado que es la base de nuestro proyecto.

4.1 Introducción

El objetivo general del proceso de minería de datos es extraer información de un conjunto de datos y transformarla en una estructura comprensible para su uso posterior [14]. Para ello utiliza métodos de inteligencia artificial, aprendizaje automático, estadística y sistemas de bases de datos.

El término es una palabra de moda, y es frecuentemente mal utilizado para referirse a cualquier forma de datos a gran escala o procesamiento de la información (recolección, extracción, almacenamiento, análisis y estadísticas), así como cualquier aplicación del sistema de apoyo a la toma de decisiones por ordenador, incluyendo la inteligencia artificial, aprendizaje automático, y la inteligencia empresarial.

La tarea de minería de datos real es el análisis automático o semi-automático de grandes cantidades de datos para extraer patrones previamente desconocidos e interesantes, tales como grupos de registros de datos (análisis de clustering), registros inusuales (detección de anomalías), y dependencias (reglas de asociación). Esto generalmente implica el uso de técnicas de bases de datos como los índices espaciales. Estos patrones se pueden ver como una especie de resumen de los datos de entrada, y se pueden usar en un análisis adicional o, por ejemplo, en aprendizaje automático y análisis predictivo.

Un proceso típico de minería de datos consta de los siguientes pasos generales:

- **Selección del conjunto de datos**, tanto en lo que se refiere a las variables objetivo (aquellas que se quiere predecir, calcular o inferir), como a las variables independientes (las que sirven para hacer el cálculo o proceso), como posiblemente el muestreo de los registros disponibles.

- **Análisis de las propiedades de los datos**, en especial histogramas, diagramas de dispersión, presencia de valores atípicos y ausencia de datos (valores nulos).

- **Transformación del conjunto de datos de entrada**, que se realiza de diversas formas en función del análisis previo, con el objetivo de prepararlo para aplicar la técnica de minería de datos que mejor se adapte a los datos y al problema, a este paso también se le conoce como *preprocesamiento* de los datos.

- **Selección y aplicación de técnicas de minería de datos**, donde se construye el modelo predictivo, de clasificación o segmentación.

- **Extracción de conocimiento**, mediante una técnica de minería de datos, se obtiene un modelo de conocimiento, que representa patrones de comportamiento observados en los valores de las variables del problema o relaciones de asociación entre dichas variables. También pueden usarse varias técnicas a la vez para generar distintos modelos, aunque generalmente cada técnica obliga a un preprocesado diferente de los datos.

Cuando se trabaja con series temporales hablamos de minería de datos de series temporales donde una serie temporal representa una colección de valores obtenidos a partir de mediciones secuenciales en el tiempo. La minería de datos de series temporales se deriva del deseo de materializar nuestra capacidad natural para visualizar la forma de los datos. Los seres humanos se basan en esquemas complejos con el fin de realizar dichas tareas. De hecho, se puede evitar el centrarse en pequeñas fluctuaciones con el fin de obtener una idea de la forma y establecer las similitudes entre los patrones casi al instante en varias escalas de tiempo. Las principales tareas relacionadas con las series temporales se describen a continuación [15]. Algunas de estas tareas se explican en profundidad más adelante. Dado que el proyecto se basa en el análisis exploratorio nos centraremos en los algoritmos de clustering y medición de distancias, así como también en métodos más específicos como son la predicción, la representación y el resumen.

- **Indexación (consultas en el contenido)**: dada una serie temporal Q y una medida de similitud/disimilitud $D(Q, C)$, encontrar la serie temporal más similar en la base de datos [16].

- **Clustering**: encontrar agrupaciones naturales de la serie temporal en la base de datos bajo alguna medida de similitud/disimilitud $D(Q, C)$ [17].

- **Clasificación**: dada una serie temporal sin etiqueta Q, asignarla a una de dos o más clases predefinidas [17]. Se conoce también como aprendizaje supervisado ya que las clases se determinan antes de examinar los datos; un conjunto de datos predefinido se utiliza en el proceso de formación y aprendizaje para reconocer patrones de interés. Los dos métodos más populares de clasificación de series temporales incluyen los arboles de decisión [18] y el de los vecinos más cercanos [19]. El rendimiento de los algoritmos de clasificación generalmente se evalúa midiendo la exactitud de la clasificación, mediante la determinación del porcentaje de objetos que están en la clase correcta. Ejemplos conocidos de aplicaciones de clasificación pueden ser: reconocimiento de imágenes y patrones, filtrado de correo no deseado, diagnóstico médico y detección de fallos en aplicaciones industriales.

- **Predicción**: dada una serie temporal Q que contiene n datos, predecir el valor en el tiempo $n + 1$.

- **Resumen**: dada una serie temporal Q que contiene n puntos de datos, donde n es un número extremadamente grande, crear una aproximación (posiblemente gráfica) de Q, que conserve sus características secuenciales esenciales, pero cabe en una sola página [20].

- **Detección de anomalías**: dada una serie temporal Q que se supone normal, y dada una serie temporal R, encontrar todas las subsecuencias de R que contienen anomalías o incidencias "sorprendentes, inesperados o interesantes" [21].

- **Segmentación**: Dada una serie temporal Q que contiene n puntos de tiempo, construir un modelo $\bar{Q}$ a partir de segmentos de la función definida a trozos K $(K \ll n)$, tal que $\bar{Q}$ esté muy próxima a Q [17].

El preprocesado de datos es el primer paso en muchos procesos de toma de decisión y de algoritmos de minería de datos. Además, hay que tener en cuenta que la indexación y la agrupación o clustering hacen un uso explícito de medidas de distancia, y muchos enfoques para la clasificación, predicción, detección de anomalías hacen un uso implícito de medidas de distancia. Por lo tanto, vamos a dedicar un apartado para explicar en detalle cada uno de los temas antes mencionados.

4.2 Preprocesado

El propósito fundamental del preprocesamiento es manipular y transformar los datos en bruto de modo que la información contenida en ellos pueda quedar expuesta para que sea más fácilmente tratable. Si los datos no se limpian y normalizan existe un alto riesgo de conseguir resultados falsos y sin sentido. Esto puede deberse a que se cuenta con un conjunto de datos inicial incompleto, con ruido o que éstos sean inconsistentes. La preparación de los datos puede generar un conjunto de datos más pequeño que el original, lo cual puede mejorar la eficiencia del proceso de minería de datos y conducir a patrones de calidad, permitiendo recuperar información incompleta o resolver conflictos [22].

El preprocesamiento de datos engloba a todas aquellas técnicas de análisis de datos que permiten mejorar la calidad de un conjunto de datos de modo que las técnicas de extracción de conocimiento o minería de datos puedan obtener mayor y mejor información. Esta actuación incluye:

- **Selección relevante de datos y limpieza:** resolviendo problemas de presentación y codificación, eliminando registros duplicados, eliminando anomalías y valores nulos e integrando datos desde diferentes tablas para crear información homogénea.

- **Reducción y transformación de datos:** seleccionando características relevantes para extraer información, discretizando o haciendo un muestreo o seleccionando instancias.

Vamos a enumerar brevemente las técnicas más importantes y dar algunos ejemplos. Un paso muy importante cuando se utilizan series temporales obtenidas a partir de mediciones de la vida real es la reducción del ruido. La reducción del ruido se puede hacer, por ejemplo, usando filtros digitales o fijando umbrales en las ondas. Los valores que faltan, a menudo pueden ser sustituidos por interpolación lineal. Las técnicas de selección de características pueden permitir mejorar la precisión e interoperabilidad de los métodos de aprendizaje automático o para proporcionar información adicional, además de reducir el tamaño de la base de datos y el tiempo de los algoritmos de aprendizaje.

El preprocesamiento de datos tiene como inconveniente el no contar con una metodología concreta de actuación para todos los problemas. Cada problema puede requerir una actuación diferente, utilizando diferentes herramientas de preprocesamiento [23].

4.2.1 Tratamiento de los valores nulos o del ruido

Los datos del mundo real suelen presentarse de forma incompleta, inconsistente y ruidosa por lo que como paso previo a su utilización hay que limpiarlos. Hay varias formas de enfrentarse a este problema:

- Rellenar los datos de manera manual, algo inabordable en la minería de datos en aplicaciones reales.
- Usar una constante global para rellenar los datos, por ejemplo, cambiar un dato ausente o que está fuera de rango por NAN (Not A Number).
- Usar la media o la moda de los valores de entorno para sustituir por el dato que falta.
- Corregir inconsistencias.
- Usar técnicas de reducción de ruido

4.2.2 Normalización de los datos

Dado que el rango de los valores de los datos en bruto puede ser muy variable, algunos algoritmos de aprendizaje automático o minería de datos no funcionan correctamente si no se normalizan los datos. Como, por ejemplo, la mayoría de los clasificadores que calculan distancias entre dos puntos mediante la distancia euclídea [24]. Si una de las dos series tiene una amplia gama de valores o un desplazamiento temporal, la distancia estará distorsionada como se puede ver en la Figura 4.1. Por lo tanto, toda función tiene que ser normalizada para que cada característica que se tenga en cuenta contribuya proporcionalmente en la distancia final. Se sabe que algunos métodos como clustering con *K*-means o el clasificador del vecino más cercano funcionan mejor si se preprocesa de esta forma los datos [25].

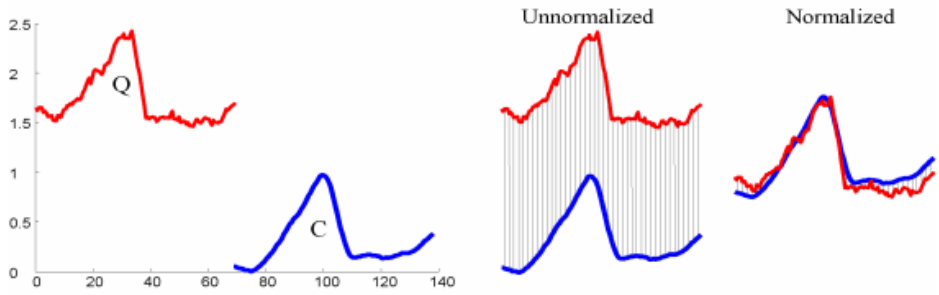


Figura 4.1: Ejemplo de series normalizadas.

A continuación, se enumeran algunos de los métodos de normalización más comunes.

- **Reescalado:** El método más sencillo es cambiar la escala de los datos y escalarlos el intervalo $[0, 1]$ o $[-1, 1]$. La elección del tipo de intervalo a usar depende de la naturaleza de los datos. La fórmula general es:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

donde:

x es el valor original y
x' es el valor normalizado

- **Normalización:** En minería de datos podemos manejar distintos tipos de datos que pueden incluir múltiples dimensiones. La normalización de los valores de cada característica nos deja valores con media cero y varianza unidad. El método general de cálculo es determinar la distribución media y la desviación estándar para cada valor. A continuación, se resta la media de cada valor. Después se dividen los valores conseguidos por la desviación estándar.

$$\boldsymbol{x' = \frac{x - \bar{x}}{\sigma}}$$

donde:

x' es el valor normalizado,
x es el valor original,
$\bar{x}$ es la media y
σ es la desviación estándar.

Este método es ampliamente utilizado para la normalización en muchos algoritmos de aprendizaje automático y minería de datos además de ser el que utilizamos en el proyecto por ser el que mejor resultado nos ha dado.

- **Escalado a longitud unidad:** Se escala los componentes de un vector de características de modo que en conjunto tienen longitud uno. Normalmente se realiza dividiendo cada componente por la longitud euclídea del vector:

$$x' = \frac{x}{\|x\|}$$

donde:

x' es el valor normalizado,
x es el valor original,
$\|x\|$ es la longitud euclídea del vector

4.3 Medidas de similitud de series temporales

La idea de similitud en series temporales es de suma importancia para muchas tareas de minería de datos como se mencionó anteriormente.

4.3.1 Distancia en series temporales

Hay cuatro categorías de medidas de similitud para series temporales numéricas.

- **Métodos basados en la forma** que comparan el aspecto general de la serie temporal.
- **Métodos basados en características** que extraen características que generalmente describen aspectos de tiempo independientes de la serie que se comparan con funciones de distancia estáticas.
- **Métodos basados en modelos** que adaptan un modelo a los datos y miden la semejanza mediante la comparación de los modelos.
- **Métodos basados en la compresión** que analizan cómo de bien dos series temporales se pueden comprimir juntas y por separado.

Estas categorías se muestran en la Figura 4.2 con algunos ejemplos explicados después.

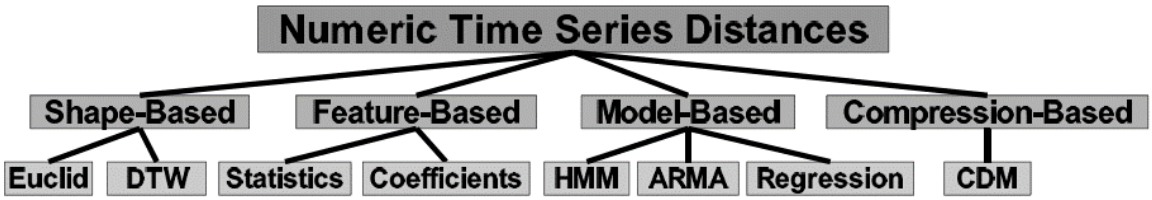


Figura 4.2: Clasificación de los distintos tipos de distancias entre series temporales numéricas.

Las series temporales cortas se comparan habitualmente por la forma. Las series temporales largas se pueden derivar y se comparan las características o modelos. Los métodos de compresión miden cómo de bien puede ser comprimida la concatenación de dos series temporales.

Explicaremos a continuación las dos distancias que usamos en el proyecto.

4.3.1.1 Distancia euclídea y distancias de Minkowski Lp

Una de las medidas de similitud más simples para series temporales es la distancia euclídea. Suponiendo que tenemos dos secuencias de la misma longitud n, podemos ver cada una como un punto en el espacio euclídeo n-dimensional, y definimos la disimilitud entre las secuencias C y Q como $D(C,Q) = Lp(C,Q)$, es decir, la distancia entre dos puntos es la distancia Lp con $p = 2$.

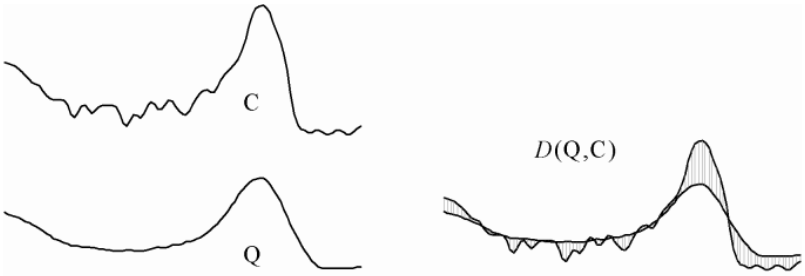


Figura 4.3: Comparación de dos series temporales mediante distancia euclídea.

Esta medida es sencilla de entender y fácil de calcular por lo que es la medida más ampliamente utilizada para la búsqueda de similitud [26]. Sin embargo, una desventaja importante de esta medida es que es muy susceptible, por ejemplo, no es útil en la situación en la que dos secuencias son iguales pero una se ha "estirado" o "comprimido" en el eje y. Este problema se puede tratar con la normalización de las secuencias antes de aplicar el operador de distancia.

Más en general, se puede utilizar la familia de distancias de Minkowski Lp poniendo énfasis en las grandes desviaciones, es decir;

Para $X = (x_1, x_2, \dots, x_n)$ y $Y = (y_1, y_2, \dots, y_n) \in \mathbb{R}^n$

Distancia Manhattan para $p = 1$

$$\left(\sum_{i=1}^{n} |x_i - y_i|^1\right)^1$$

Distancia euclídea para $p = 2$

$$\left(\sum_{i=1}^{n} |x_i - y_i|^2\right)^{1/2}$$

Distancia de Chebyshev para $p = \infty$

$$\lim_{p\to\infty}\left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{1/p} = \max_{1\le i\le n}|x_i - y_i|$$

Hay varios aspectos a tener en cuenta si se usan las distancias basadas en espacios Lp: la escala y la transformación de las amplitudes o el eje temporal, el ruido y los valores atípicos, el alineamiento temporal uniforme y no uniforme. Por lo tanto, muchos autores han utilizado transformaciones euclídeas que hacen que la elección de las transformaciones para obtener una medida de distancia entre series temporales sea más significativa.

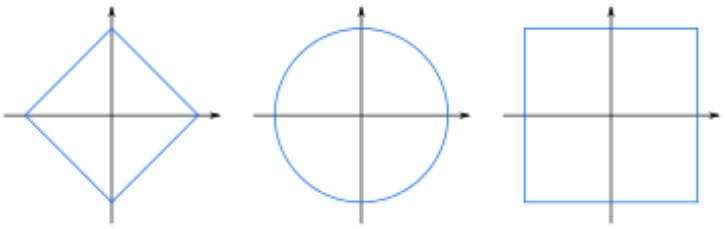


Figura 4.4: Distancias Minkowski para $p = 1$ (Manhattan), $p = 2$ (Euclídea) y $p = \infty$ (Chebyshev).

Una transformación o escalado de la amplitud puede producir una gran distancia entre dos series, incluso si tienen una forma similar. Un ejemplo común donde no se desea esto es en la comparación de precios de acciones. El valor absoluto de una población general no es tan interesante como la forma de movimientos ascendentes y descendentes. Este problema se puede superar mediante una transformación lineal o normalización de las amplitudes [27]. Es necesario aplicar a ambas series

una normalización a un rango fijo [28] o media cero y varianza unidad [29]. Estos métodos son rápidos y muy comunes, pero no dan una coincidencia óptima de dos series bajo transformaciones lineales.

Una transformación lineal solo se debe aplicar a una de las series temporales, porque de haber dos transformaciones al mismo tiempo siempre resulta en factores de escala cero como la respuesta óptima (e inútil). El factor de escala, en particular, no puede ser negativo para muchas aplicaciones ya que esto corresponde a una forma invertida.

Tanto la transformación lineal como la normalización deben ser manejados con cuidado en presencia de ruido. Si una serie temporal es esencialmente plana con un poco de ruido, normalizar a varianza unitaria hará que se vea muy distinta. El ruido también se puede acentuar por transformaciones lineales con el fin de que coincida con las características de otra serie temporal.

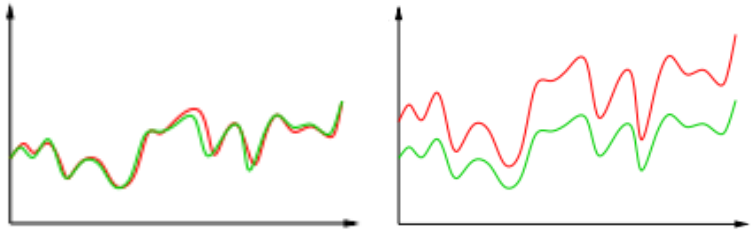


Figura 4.5: Secuencias que coinciden aproximadamente y transformación de escala

El escalado y transformación del tiempo no suele ser un problema debido a que los valores de los ejes temporales no son usados normalmente en el cálculo de las distancias, sino que se usa el orden de los valores. Sin embargo, los problemas surgen con series temporales de diferentes longitudes y/o frecuencias de muestreo. El remuestreo puede ser utilizado para hacer que dos series temporales tengan la misma longitud y a continuación, aplicar una función de distancia, a veces llamada *alineamiento temporal uniforme* [30]. Se ha observado que la reducción de muestreo de la serie más larga es más eficiente y robusta frente al ruido.

Las pequeñas distorsiones del eje temporal se tratan comúnmente con *alineamiento temporal no uniforme*, más precisamente con el *alineamiento temporal dinámico* (*Dynamic Time Warping*, DTW) que se verá en la siguiente sección.

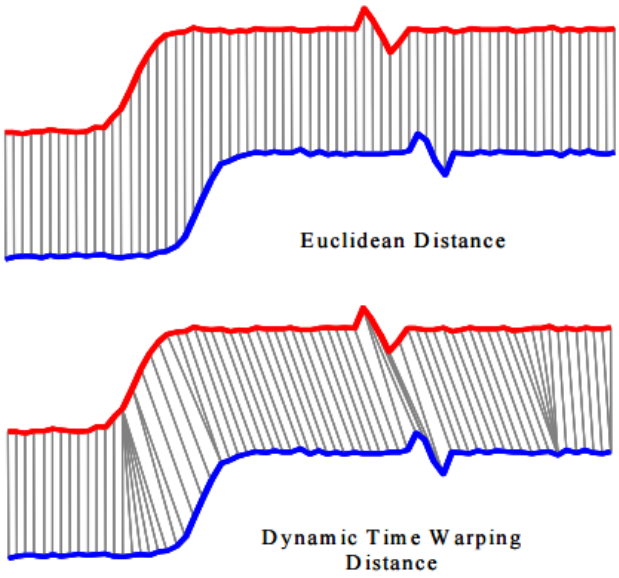


Figura 4.6: Aplicación de la distancia euclídea y DTW a dos series.

En el ejemplo de la Figura 4.6 se presentan dos series con una forma similar que no están alineadas en el eje temporal. Se puede ver que la distancia euclídea asume que el punto i-ésimo de la primera serie coincide o está alineado con el mismo punto i-ésimo de la segunda serie, produciendo así una medida de disimilitud pesimista. Mientras que la alineación temporal dinámica (DTW) no lineal permite calcular una medida de distancia más adecuada.

4.3.1.2 Alineamiento temporal dinámico (Dynamic Time Warping)

DTW es un método que calcula una coincidencia óptima entre dos secuencias dadas en las que se observa una variabilidad interna en la duración de los puntos que la forman, de modo que no existe una sincronización o alineamiento temporal. Además, esta falta de sincronización no obedece a una ley fija, por ejemplo, un retardo constante, sino que se da de forma heterogénea, produciéndose así variaciones localizadas que aumentan o disminuyen la duración del tramo de análisis.

La problemática asociada hace referencia a la dificultad añadida en el proceso de medida de distancia entre patrones, puesto que se estarán comparando tramos que pueden corresponder a grupos de puntos distintos. Será necesario alinear temporalmente la serie para proceder a realizar una medida de distancia entre patrones cuyo nuevo eje temporal haya homogeneizado las variaciones iniciales. Este método de alineación se utiliza a menudo en la clasificación de series temporales.

Para llevar a cabo el alineamiento el eje temporal de la serie se comprime y expande de forma no lineal para alinear los vectores de características entre las dos series. Fruto de este proceso surge lo que se conoce como camino de alineamiento. La alineación optima se encuentra mediante el cálculo del camino de alineamiento más corto en la matriz de distancias entre todos los pares de puntos de tiempo bajo una serie de limitaciones. Se sabe también que el DTW es más preciso que la distancia euclídea para la clasificación y clustering o agrupamiento de series temporales.

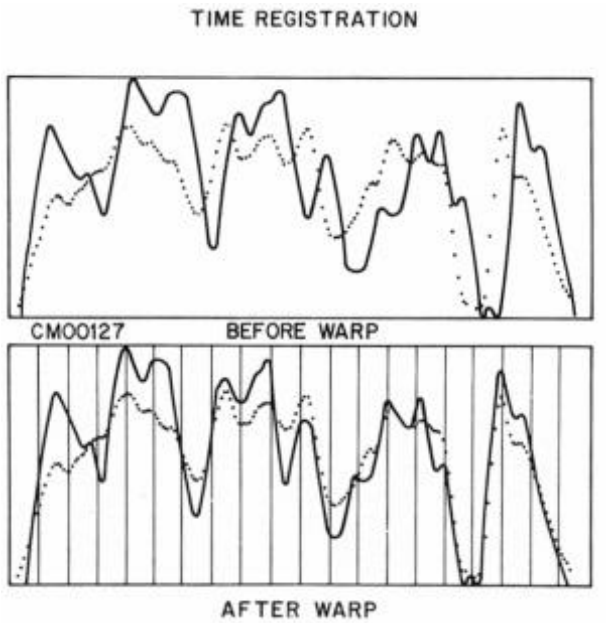


Figura 4.7: Ejemplo de alineación con DTW

Suponiendo que tenemos dos series temporales, una secuencia Q de longitud n y una secuencia C de longitud m, donde:

$$Q = q_1, q_2, \dots, q_i, \dots, q_n$$
$$C = c_1, c_2, \dots, c_i, \dots, c_m$$

Para alinear estas dos secuencias usando DTW se construye una matriz M de $n \times m$ donde el (i-ésimo, j-ésimo) elemento m_{ij} corresponde a la distancia cuadrática, $d(q_i, c_j) = (q_i - c_j)^2$, que es el alineamiento entre los puntos q_i y c_i. Para encontrar la mejor coincidencia entre las dos secuencias se busca un camino a través de la matriz que minimice el total acumulado de distancias entre puntos. La distancia entre puntos que se suele usar es la euclídea o Manhattan, ésta última es la que usamos nosotros en el algoritmo DTW.

Un camino, W, es un conjunto contiguo de elementos de la matriz M que caracteriza a un mapeo entre Q y C. El k-ésimo elemento de W se define como $w_k = (i,j)_k$. Por lo que se obtiene:

$$W = w_1, w_2, \dots, w_k, \dots, w_K \qquad \text{donde} \quad \max(m,n) \leq K < m + n - 1$$

Por definición, el camino óptimo W_o es el camino que minimiza el coste de alineamiento.

$$DTW(Q,C) = \left\{ \sqrt{\sum_{k=1}^{K} w_k} \right\}$$

Este camino se puede encontrar utilizando programación dinámica para evaluar la siguiente recurrencia que define la distancia acumulada $\gamma(i,j)$ como la distancia $d(i,j)$ encontrada en la celda actual y el mínimo de las distancias acumuladas de los elementos adyacentes:

$$\gamma(i,j) = d(q_i, c_j) + \min\{ \gamma(i-1, j-1), \gamma(i-1, j), \gamma(i, j-1) \}$$

En la práctica, no se evalúan todos los posibles caminos de alineamiento. En su lugar, se tienen en cuenta las siguientes limitaciones que reducen el número de caminos considerados durante el proceso de correspondencia. Esta reducción del número de caminos consideras también tiene el efecto secundario deseable de acelerar los cálculos, aunque solo en un (pequeño) factor constante.

- **Condiciones de contorno:** El camino debe empezar en $w_1 = (1,1)$ y terminar en $w_K = (m,n)$, es decir, el camino de alineamiento tiene que empezar en la parte inferior izquierda y terminar en la parte superior derecha de la matriz.

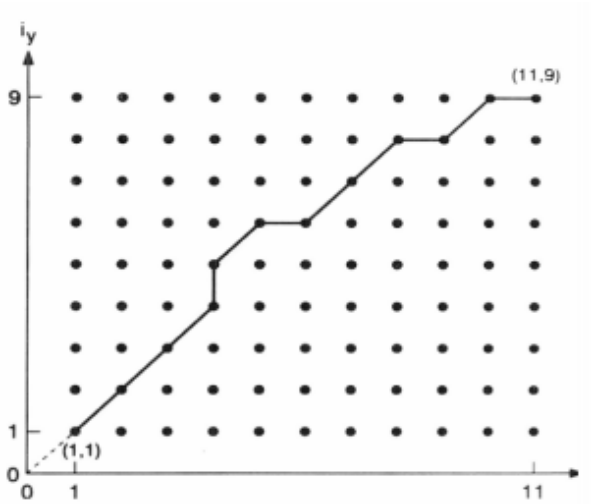


Figura 4.8: Ejemplo de condición de contorno.

- **Condición de continuidad:** Cada punto en la lista de candidatos debe ser usado en el camino de alineamiento y ambos índices i y j solo pueden aumentar en 0 o 1 en cada paso del camino. En otras palabras, si se toma un punto (i, j) de la matriz, el punto anterior debe haber sido $(i-1, j-1)$, $(i-1, j)$ o $(i, j-1)$.

- **Condición de monotonía:** Dado $w_k = (a, b)$ entonces $w_{k-1} = (a', b')$ donde $a - a' \geq 0$ y $b - b' \geq 0$. El camino de alineamiento no puede ir hacia atrás en el tiempo; ambos índices i y j cualesquiera permanecen igual o aumentan. Nunca pueden disminuir.

- **Condición de restricción de pendiente:** El camino no debe ser demasiado empinado o demasiado poco profundo. Esto evita que subsecuencias muy cortas coincidan con otras muy largas. La condición se expresa como una relación a / b, donde b es el número de pasos en la dirección x y a es el número de pasos en la dirección y. Tras b pasos en x, se debe dar un paso en y, y viceversa.

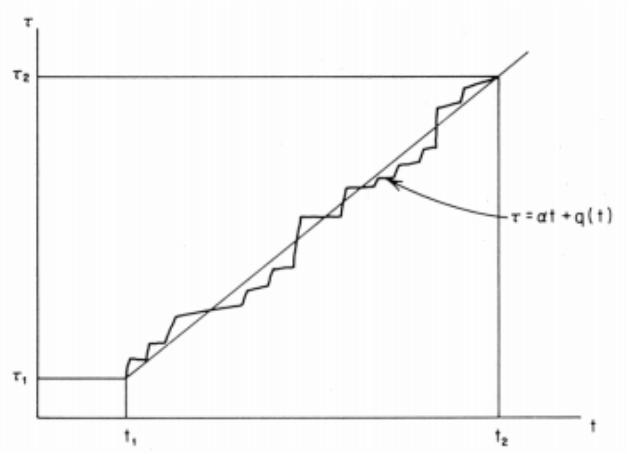


Figura 4.9: Ejemplo de restricción de la pendiente.

- **Condición de la ventana de ajuste:** Un camino de alineamiento intuitivo es poco probable que derive muy lejos de la diagonal. La distancia que se permite que el camino recorra se limita a una ventana de tamaño r, directamente por encima y a la derecha de la diagonal.

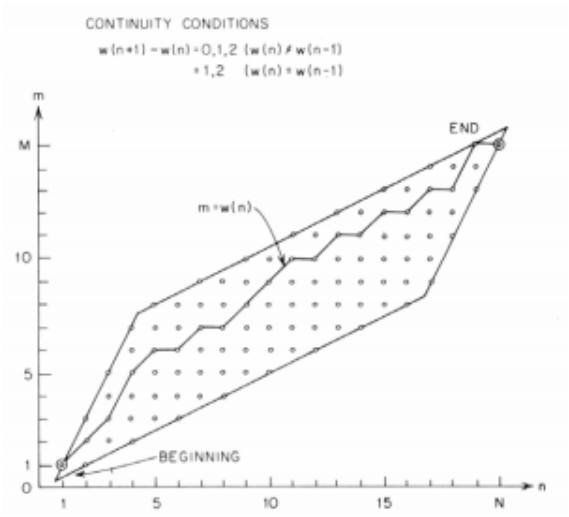


Figura 4.10: Ejemplo de ventana de ajuste y continuidad.

Mediante la aplicación de estas condiciones, se pueden restringir los movimientos que se pueden hacer desde cualquier punto del camino y, por tanto, se reducen el número de caminos que deben tenerse en cuenta.

La distancia euclidiana entre dos secuencias se puede ver como un caso especial de DTW donde el k-ésimo elemento de W está limitado de tal manera que $w_k = (i,j)_k$, $i = j = k$. Hay que tener en cuenta que solo se define en el caso especial de que las dos secuencias tienen la misma longitud. La complejidad temporal y espacial de DTW es de $O(nm)$ [31]. Sin embargo, las restricciones anteriores mitigan este coste en un factor constante.

4.4 Clustering

De todas las técnicas aplicadas a las series temporales, el clustering o agrupación es quizás la más utilizada, siendo útil en sí mismo como una técnica de exploración, y como una subrutina en otros algoritmos más complejos de minería de datos [32]. Se puede considerar el problema de aprendizaje no supervisado más importante. Una gran parte de la atención de la comunidad de minería de datos se centra en los datos de series temporales. Esto es comprensible y muy esperado ya que los datos de series temporales son un subproducto en prácticamente todas las actividades humanas, incluyendo la biología [33], las finanzas, la geología, la exploración del espacio, la robótica y el análisis del movimiento humano.

Una definición de clustering podría ser "el proceso de organización de objetos en grupos cuyos miembros son similares de alguna manera". Por tanto, un clúster o grupo es una colección de objetos que son "similares" entre ellos y son "diferentes" a los objetos que pertenecen a otros grupos. Podemos demostrar esto con un ejemplo gráfico sencillo. En la Figura 4.11 se identifican fácilmente los cuatro grupos en los que los datos se pueden dividir; el criterio de similitud es la distancia donde dos o más objetos pertenecen al mismo grupo si están "cerca" de acuerdo a una distancia determinada (en este caso la distancia euclídea). Esto se llama *agrupación basada en distancia*. Otro tipo de agrupación es el agrupamiento conceptual donde dos o más objetos pertenecen al mismo grupo si éste define un concepto común a todos los demás objetos. En otras palabras, los objetos se agrupan en función de su ajuste a los conceptos descriptivos, no de acuerdo a las medidas de similitud simples [34].

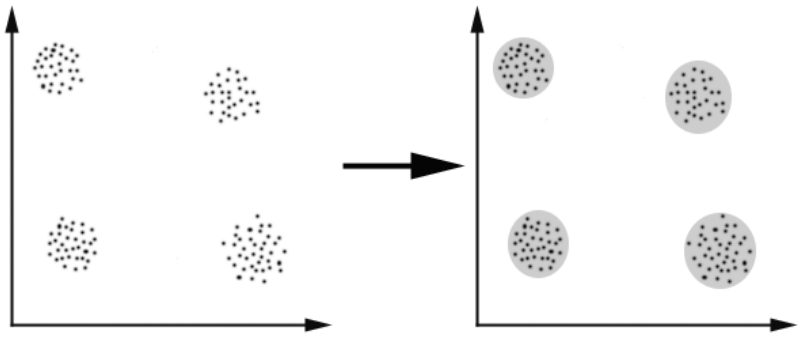


Figura 4.11: Ejemplo de clustering.

4.4.1 Las metas del clustering

El clustering[8] es el proceso de crear grupos de datos según un criterio de distancia o similitud. Cada grupo debe ser lo más homogéneo posible y distinto de otros grupos. Es una técnica exploratoria cuyo objetivo es determinar la agrupación instrínseca en un conjunto de datos sin etiqueta. Pero, ¿cómo decidir lo que constituye una buena agrupación? Se puede demostrar que no hay un criterio absoluto que resulte mejor independientemente del objetivo final de la agrupación. En consecuencia, es el usuario quien debe proporcionar este criterio, de tal manera que el resultado de la agrupación se adapte a sus necesidades. Por ejemplo, podríamos estar interesados en la búsqueda de representantes de grupos homogéneos (reducción de datos), en la búsqueda de *grupos naturales* y describir sus propiedades desconocidas (tipo de datos *naturales*), en la búsqueda de agrupaciones útiles y adecuadas (tipo de datos "útiles") o en la búsqueda de objetos de datos inusuales (detección de valores atípicos).

[8] A veces clustering recibe también el nombre de clasificación sin supervisión.

4.4.2 Posibles aplicaciones

Los algoritmos de clustering se pueden aplicar en muchos campos, por ejemplo:

- Comercialización: búsqueda de grupos de clientes con un comportamiento similar dado en una gran base de datos de clientes que contienen sus propiedades y registros de compra pasados.
- Biología: clasificación de plantas y animales dadas sus características.
- Seguros: identificación de grupos de usuarios de seguros de motor con un costo promedio elevado: la identificación de fraudes.
- Urbanístico: identificación de grupos de viviendas de acuerdo a su tipo de casa, el valor y la ubicación geográfica.
- Estudio de terremotos: observación de epicentros sísmicos con clustering para identificar zonas peligrosas

4.4.3 Requisitos

Los principales requisitos que un algoritmo de clustering debe cumplir son:

- Escalabilidad
- Tratar bien con diferentes tipos de atributos que haya
- El descubrimiento de agrupaciones con forma arbitraria
- Conocimientos mínimos del dominio para determinar los parámetros de entrada
- Habilidad para tratar con el ruido y los valores extremos
- Insensibilidad al orden de los registros de entrada
- Alta dimensionalidad
- Interpretabilidad y facilidad de uso

4.4.4 Problemas

Hay varios problemas con el clustering. Entre ellos se puede encontrar que las técnicas actuales de clustering no atienden adecuadamente todos los requisitos, al mismo tiempo que tratar con un gran número de dimensiones y un gran número de datos puede ser complicado debido a la complejidad. La eficacia del método utilizado depende de la definición de distancia (para el agrupamiento basado en distancia); si no existe una medida de distancia obvia se tiene que definir, lo cual no siempre es fácil, especialmente en espacios multidimensionales. Por último, el resultado del agrupamiento, que en muchos casos puede ser en sí arbitrario, puede interpretarse de diferentes maneras.

4.4.5 Algoritmos de clustering

La idea de clúster o grupo no se puede definir de manera unívoca ya que puede obedecer a distintos criterios, y esto es una de las razones por las que hay tantos algoritmos de clustering. Hay un denominador común: un grupo de datos. Sin embargo, se pueden usar diferentes modelos de clúster, y para cada uno de estos modelos se pueden dar de nuevo diferentes algoritmos [35].

Así, la idea de clúster varía significativamente en sus propiedades según los diferentes algoritmos. La comprensión de estos modelos es clave para entender las diferencias entre los distintos algoritmos. El algoritmo apropiado para un problema particular a menudo tiene que ser elegido de forma experimental, a menos que exista una razón matemática para preferir un modelo sobre otro [36].

Los modelos típicos de clúster incluyen:

- **Modelos de conectividad:** por ejemplo, la agrupación jerárquica construye modelos basados en la distancia de las conexiones.
- **Modelos centroide:** por ejemplo, el algoritmo *K*-means representa cada grupo por un único vector medio.
- **Modelos de densidad:** por ejemplo, DBSCAN[9] y OPTICS[10] definen grupos como regiones densas conectadas en el espacio de los datos.
- **Modelos de grupo:** algunos algoritmos no proporcionan un modelo refinado para sus resultados y solo proporcionan la información de la agrupación.
- **Modelos basados en grafo:** un clique, es decir, un subconjunto de nodos en un grafo tal que cada dos nodos en el subconjunto están conectados por una arista puede ser considerado como un prototipo o forma de grupo.

4.4.5.1 Clustering basado en conectividad o agrupamiento jerárquico

El agrupamiento jerárquico está basado en la idea principal de que los objetos más cercanos están más relacionados que los que están más alejados. Estos algoritmos conectan elementos para formar los grupos basándose en la distancia. Un grupo puede ser descrito, en gran parte, por la distancia máxima necesaria para conectar todos los elementos del grupo. A distancias diferentes, se formarán grupos diferentes, los cuales pueden ser representados usando un dendrograma que explica de dónde proviene el nombre de agrupamiento jerárquico. Estos algoritmos no solo proporcionan una única partición del conjunto original de datos, sino que devuelve una amplia jerarquía de grupos que se unifican unos con otros a ciertas distancias. En un dendrograma, el eje y marca la distancia a la que los grupos se unifican, mientras que los objetos se colocan a lo largo del eje x de tal manera que los grupos no se mezclan. Se puede ver un ejemplo en la Figura 4.12.

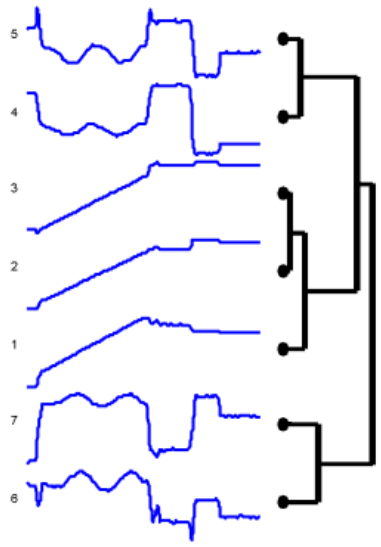


Figura 4.12: Ejemplo de dendrograma.

[9] Agrupamiento espacial basado en densidad de aplicaciones con ruido.
[10] Ordenación de puntos para identificar la estructura de clustering.

El agrupamiento basado en conectividad es una familia entera de métodos que difiere en cómo se miden las distancias. Aparte de la elección habitual de funciones de distancia, el usuario también tiene que decidir sobre el criterio de vinculación para su uso ya que un grupo se compone de varios objetos y hay varios candidatos hacia los que calcular la distancia. Las estrategias para el agrupamiento jerárquico generalmente son de dos tipos:

- **Aglomerativas:** que consiste en un enfoque ascendente, cada observación comienza en su propio grupo y los pares de grupos son mezclados mientras se sube en la jerarquía.

  A continuación, enumeramos algunas técnicas de encadenamiento.

  - **CLINK:** en el *agrupamiento de enlace completo* se fusionan en cada paso los dos grupos cuya fusión tiene el diámetro más pequeño o las dos agrupaciones con la distancia máxima por parejas más pequeña [37].

    $$D(X,Y) = \max_{x \in X, y \in Y} d(x,y)$$

    donde:

    X, Y son dos grupos de elementos y
    $d(x,y)$ es la distancia entre los elementos $x \in X$ y $y \in Y$

  - **SLINK:** en el *agrupamiento de enlace simple* se fusionan en cada paso los dos grupos cuyos dos miembros más cercanos tienen la menor distancia o las dos agrupaciones con la distancia mínima por parejas más pequeña [37].

    $$D(X,Y) = \min_{x \in X, y \in Y} d(x,y)$$

    donde:

    X, Y son dos grupos de elementos y
    $d(x,y)$ es la distancia entre los elementos $x \in X$ y $y \in Y$

    Se puede ver un ejemplo en la Figura 4.13 donde se crean 35 grupos, al principio el grupo más grande se fragmenta en grupos más pequeños, mientras que todavía está conectado al segundo mayor por el efecto de enlace simple.

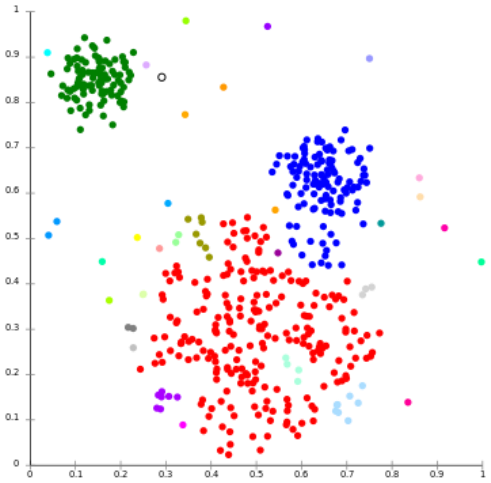


Figura 4.13: Ejemplo de agrupamiento con SLINK en datos Gaussianos.

- **Agrupamiento de promedios:** hay un compromiso entre el agrupamiento de valores extremos de CLINK y la tendencia a formar largas cadenas de elementos de SLINK. En el proyecto usamos el algoritmo **UPGMA**[11] que construye un dendrograma que refleja la estructura presente en una matriz de similitud por pares. En cada iteración los dos grupos más cercanos se combinan en un grupo que se encuentra en un nivel superior [38].

$$\overline{D}(A,B) = \frac{1}{|A| \cdot |B|} \sum_{x \in A} \sum_{y \in B} d(x,y)$$

donde:

A y B son dos grupos de elementos de cardinal $|A|$ y $|B|$ respectivamente y $d(x,y)$ es la distancia promedio entre los elementos $x \in A$ y $y \in B$

- **Divisivas:** que consiste en un enfoque descendente donde todas las observaciones comienzan en un grupo y se realizan divisiones mientras se va bajando en la jerarquía. Este procedimiento se conoce como algoritmo DIANA[12] y se aplica de forma recursiva hasta que cada elemento forma en sí mismo un grupo [39].

  El agrupamiento descendente es conceptualmente más complejo que el ascendente ya que hace falta un segundo algoritmo que se usa como subrutina para dividir el grupo inicial, pero tiene la ventaja de ser más eficiente si no se genera una jerarquía completa, es decir, si no se llega a las hojas del árbol. Para un número fijo de niveles se puede usar como subrutina el algoritmo *K*-means que tiene un coste lineal respecto al número de elementos y grupos [40].

En el caso general, la complejidad del agrupamiento aglomerativo es $O(n^3)$, lo cual lo hace demasiado lento para grandes conjuntos de datos. En cambio, el agrupamiento divisivo con búsqueda exhaustiva tiene una complejidad de $O(2^n)$, lo cual es todavía peor. Sin embargo, para algunos casos especiales, se conocen métodos aglomerativos eficientes y óptimos de complejidad $O(n^2)$ como SLINK para agrupamientos de enlace simple y CLINK para agrupamientos de enlace completo y de complejidad $O(n^2 \log n)$ como UPGMA para agrupamientos de promedios.

4.4.5.2 Clustering basado en centroide

En el agrupamiento basado en centroide los grupos se representan por un vector central que puede no ser necesariamente un miembro del conjunto de datos. Cuando el número de grupos se fija a k, el algoritmo *K*-means [41] da una definición formal como un problema de optimización: se trata de encontrar los k centros de los grupos y asignar los objetos al centro del grupo más cercano, de manera que las distancias al cuadrado del grupo al centro estén minimizadas.

Este problema de optimización es NP-Completo, y por ello el objetivo común es buscar solo soluciones aproximadas. Un método aproximado particularmente bien conocido es el algoritmo de Lloyd [42], al que se hace referencia normalmente como algoritmo *K*-means [41]. Sin embargo, solo encuentra un óptimo local y normalmente es necesario ejecutarlo varias veces con diferentes inicializaciones aleatorias.

- **Algoritmo *K*-means:** Es uno de los algoritmos de aprendizaje sin supervisión más simples que resuelven el problema de agrupación. El procedimiento sigue una manera simple y fácil de clasificar un determinado conjunto de datos a través de un cierto número de grupos

[11] Unweighted Pair Group Method with Arithmetic Mean
[12] DIvisive ANAlysis Clustering

(suponiendo *k* grupos) fijados a priori. La idea principal es definir *k* centroides, uno por cada grupo. Estos centroides deben colocarse de manera astuta ya que ubicaciones diferentes provocan resultados diferentes. Por lo tanto, la mejor opción es colocarlos lo más alejados posible de cualquier otro centroide. El segundo paso es tomar cada del conjunto de datos y asociarlo con el centroide más cercano. Cuando ya no quedan puntos por asignar se ha finalizado el segundo paso obteniéndose así un primer agrupamiento. En este punto, como tercer paso hay que volver a calcular *k* nuevos centroides como baricentros de las agrupaciones resultantes del paso anterior. Después con estos nuevos centroides se vuelve a hacer el segundo paso asignando los puntos al nuevo centroide más cercano. Los pasos segundo y tercero se repiten hasta que los centroides dejan de variar su posición [43].

En el ejemplo de la Figura 4.14 se eligen inicialmente tres centroides (círculos azul, verde y rojo) que cambian su posición al terminar de aplicar el algoritmo. Los grupos creados inicialmente también varían al cambiar la posición de los centroides.

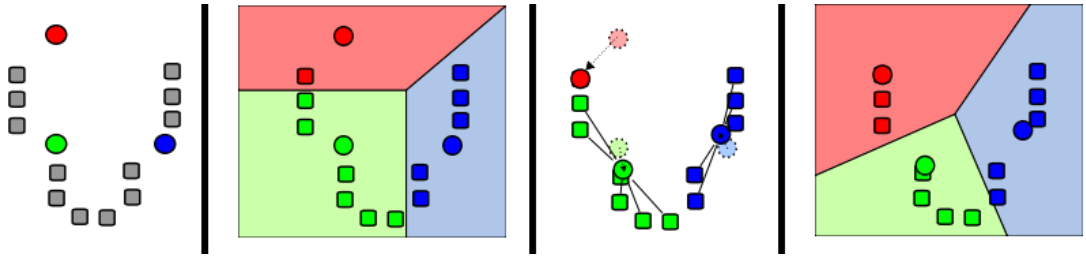


Figura 4.14: Ejemplo de K-means.

K-means tiene una serie de propiedades teóricas interesantes [44]. En primer lugar, divide el espacio de datos en una estructura conocida como diagrama de Voronoi, donde se supone igual tamaño para los grupos. En segundo lugar, es conceptualmente cercano al método de clasificación del vecino más cercano, y como tal es muy popular en el aprendizaje automático. En tercer lugar, se puede ver como una variación de la clasificación basada en modelos.

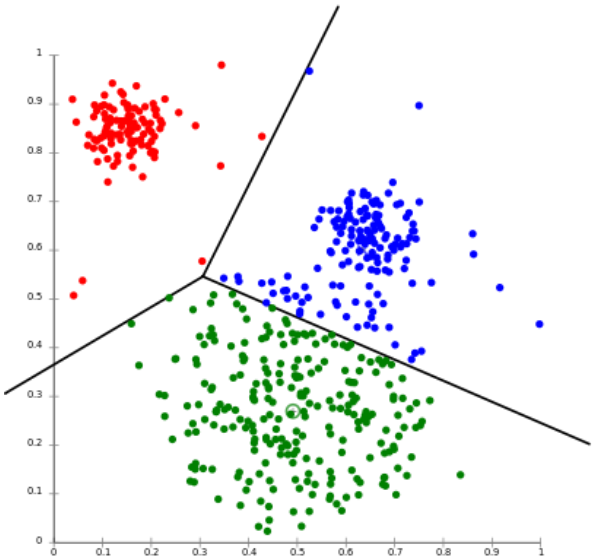


Figura 4.15: Esquemas de Voronoi resultantes de ejecutar K-means

Las variaciones de *K*-means a menudo incluyen optimizaciones tales como la elección de la "mejor" de las múltiples ejecuciones, donde una forma de elegir una mejor solución es con el método Elbow, que trata de elegir el número de grupos obtenidos de forma que al añadir otro grupo no se obtiene una mejora significativa [45]. En la Figura 4.16 se muestran dos funciones como ejemplo de aplicación del método Elbow, en la función de la izquierda se aprecia que elegir la ejecución que

genera tres grupos es la que mejores resultados obtiene ya que usar cuatro grupos no presenta ninguna mejora importante. Por el contrario, en la función de la derecha podría usarse un $k = 3$ pero también un $k = 4$ dado que no se aprecia demasiada diferencia. Otras variaciones incluyen el restringir el centroide a miembros del conjunto de datos, elegir medianas, elegir los centros iniciales menos al azar (K-means++) o permitir una asignación de clústers difusa (Fuzzy c-means).

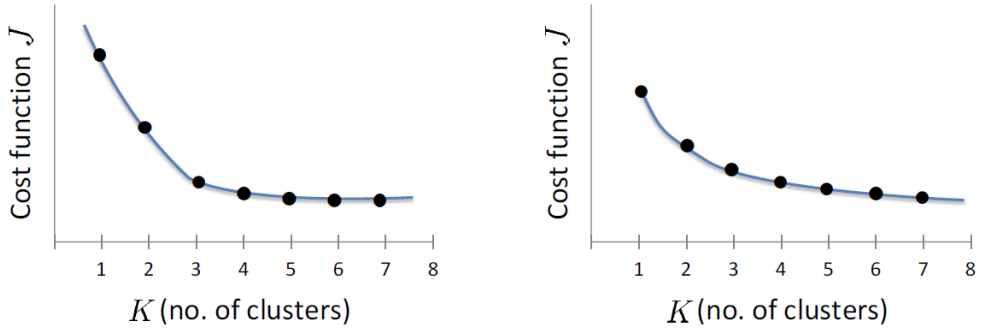


Figura 4.16: Ejemplo de método Elbow.

4.4.5.3 Clustering basado en distribuciones

El modelo de Clustering más estrechamente relacionado con la estadística se basa en modelos de distribución. Los grupos pueden ser fácilmente definidos como los objetos que pertenecen más probablemente a la misma distribución. Una propiedad conveniente de esta aproximación es que esto se parece mucho a la manera en la que los conjuntos de datos artificiales están generados: por muestreos aleatorios de objetos de una distribución.

Mientras que el fundamento teórico de estos métodos es excelente, sufren del problema clave conocido como sobreajuste o overfitting, a menos que las restricciones estén incluidas en la complejidad del modelo. Un método destacado se conoce como los modelos de mezcla Gaussianos (usando el algoritmo esperanza-maximización [46] o algoritmo EM). En este caso, el conjunto de datos se suele representar mediante un número fijo (para evitar el sobreajuste) de distribuciones Gaussianas o distribuciones normales que se inicializan al azar y cuyos parámetros se han optimizado de manera iterativa para ajustarse mejor al conjunto de datos. Este convergerá a un óptimo local, por lo que varias ejecuciones pueden producir diferentes resultados.

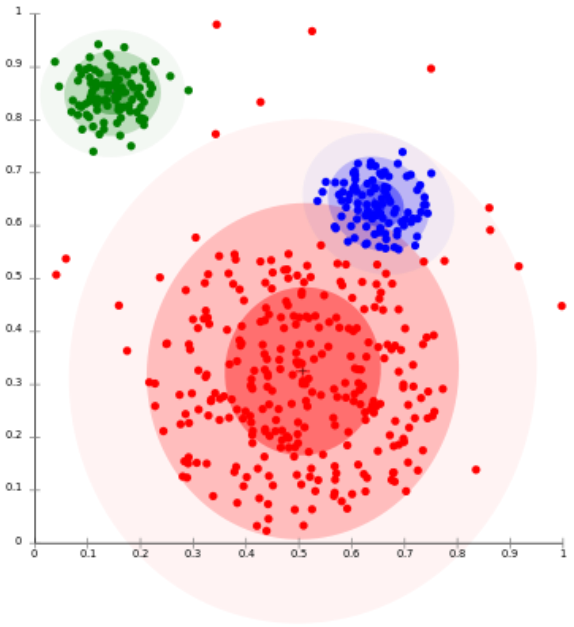


Figura 4.17: EM usado en datos de una distribución normal.

4.4.5.4 Clustering basado en densidad

Los grupos, en este caso, se definen como áreas de densidad mayor que el resto del conjunto de datos. Los elementos en estas áreas dispersas, necesarias para separar grupos, por lo general se consideran puntos de ruido y frontera. El método más popularmente conocido es DBSCAN[13] [47]. En contraste con muchos métodos más nuevos, presenta un modelo de grupo bien definido llamado *densamente alcanzable*. Similar al agrupamiento basado en conectividad, se basa en conectar puntos dentro de cierto umbral de distancia. Aun así, solo conecta aquellos puntos que satisfacen un criterio de densidad, en la variante original definida como número mínimo de otros objetos dentro de un radio dado.

Otra propiedad interesante de DBSCAN es que su complejidad es bastante baja y que llegará esencialmente los mismos resultados (es determinista para núcleos y puntos de ruido, pero no para puntos de frontera) en cada ejecución, por tanto, no hay necesidad de ejecutarlo varias veces.

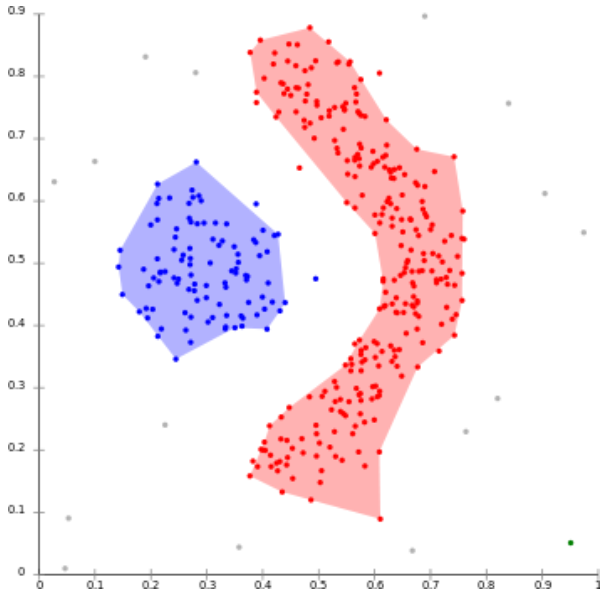


Figura 4.18: Ejemplo de agrupamiento basado en densidad con DBSCAN.

[13] Agrupamiento espacial basado en densidad de aplicaciones con ruido.

4.5 Predicción

La predicción por lo general se entiende como la previsión de los próximos valores de una serie temporal numérica o simbólica con la mayor precisión posible, teniendo en cuenta toda información disponible. La predicción puede ser vista como un tipo de clustering o clasificación. La diferencia es que la predicción es la previsión de un estado futuro, en lugar de uno actual. Sus aplicaciones incluyen la obtención de advertencias de desastres naturales (inundaciones, huracanes, tormentas de nieve, etc.), predicción de epidemias, entre otros. Muchas aplicaciones de predicción de series temporales se pueden ver en dominios económicos.

Se utilizan valores conocidos para predecir valores futuros en base a las tendencias históricas y estadísticas. Por ejemplo, con el auge de los mercados competitivos de energía, la predicción de la electricidad se ha convertido en una parte esencial de la planificación de un sistema de energía eficiente. Esto incluye la predicción de futuras demandas de electricidad basados en datos históricos y otra información como, por ejemplo, la temperatura, los precios, etc.

La meta es lo que sería deseable que suceda. Los objetivos deben estar vinculados a las previsiones y planes, pero esto no siempre ocurre. Con demasiada frecuencia, las metas se fijan sin ningún plan para la manera de alcanzarlos y sin predicciones de realismo. La planificación es una respuesta a las previsiones y objetivos, implica determinar acciones apropiadas que se requieran para hacer que las predicciones coincidan con los objetivos [48].

Para la predicción de series temporales hay una enorme cantidad de literatura, sobre todo en el campo de la estadística. Los métodos más comunes para la modelización de series temporales y predicción son los modelos ARIMA[14] de procesos estacionarios y modelos GARCH[15] de procesos con varianza no estacionaria. Además, se han propuesto muchas técnicas para aumentar la precisión de la predicción de series temporales, incluyendo las técnicas de redes neuronales y el uso de la reducción de dimensionalidad [49].

4.5.1 Alisado exponencial

El alisado exponencial se propuso a finales de 1950 y ha motivado algunos de los métodos de pronósticos de mayor éxito [50]. Las previsiones obtenidas usando métodos de suavizado exponencial son promedios ponderados de las observaciones del pasado, con los pesos en descomposición exponencial a medida que las observaciones se hacen mayores. En otras palabras, la observación más reciente es la que tendrá un peso asociado más alto. Este sistema genera una predicción fiable de forma rápida y para una amplia gama de series temporales, lo que es una ventaja y de gran importancia para muchas aplicaciones.

La forma más simple de alisado exponencial se da por la fórmula:

$$s_t = \alpha \cdot x_t + (1 - \alpha) \cdot s_{t-1}$$

donde:

α es el factor de alisamiento y $0 < \alpha < 1$.

En otras palabras, el valor alisado s_t es un simple promedio ponderado de la observación actual x_t y el valor alisado anterior s_{t-1}. Los valores de α cercanos a uno tienen menos efecto suavizante y

[14] Modelo autorregresivo integrado de media móvil.
[15] Heterocedasticidad condicional autorregresiva generalizada.

dan mayor peso a los cambios recientes de los datos, mientras que los valores de α cercanos a cero tienen un mayor efecto suavizante y son menos sensibles a los cambios recientes. En el caso límite, con $\alpha = 1$, la serie resultante es la misma que la serie original. No existe un procedimiento único para la correcta elección de α. A veces, se utilizan criterios estadísticos para elegir un factor apropiado. Alternativamente, se puede utilizar una técnica estadística para optimizar el valor de α. Por ejemplo, el método de los mínimos cuadrados se puede usar para determinar el valor de α para el que se minimiza $(s_{n-1} - x_{n-1})^2$.

Este suavizado exponencial es fácilmente aplicable y produce valores estadísticos alisados tan pronto como dos observaciones estén disponibles.

4.6 Resumen

Teniendo en cuenta que los datos de series temporales pueden ser enormemente largos, un resumen de éstos puede ser útil y necesario. Un resumen estadístico de los datos, como la media u otras propiedades estadísticas se pueden calcular fácilmente a pesar de que podría no ser particularmente valiosa la información extraída. Por el contrario, a menudo podemos utilizar el lenguaje natural, la visualización o una representación gráfica para extraer información útil o significativa a partir de los datos. La detección de anomalías y descubrimiento del "motivo" son casos especiales de resumen donde los patrones anómalos o los patrones repetitivos son de interés y informativos.

El resumen también puede ser visto como un tipo especial de problema de agrupamiento que mapea datos en subconjuntos con descripciones sencillas asociadas en forma de texto o gráfico y proporciona una visión a un nivel superior de los datos. Esta nueva descripción simplificada de los datos se utiliza en lugar de todo el conjunto de datos. El resumen se puede realizar en múltiples granularidades y para diferentes dimensiones [15].

4.6.1 Representación

Las series temporales suelen tener gran longitud lo que hace necesario usar mucho espacio en disco para almacenarlas. En la actualidad el almacenamiento no genera ningún problema, pero sí lo hace el hecho de que muchas técnicas de minería de datos gastan la mayoría de su tiempo en traer datos de disco a memoria principal para procesarlos buscando información útil. Por lo tanto, la principal motivación de representar una serie temporal de alguna forma que no sea usando los valores reales es para representar mejor las características principales de los datos de una manera concisa. Las ventajas adicionales pueden ser de compresión y por lo tanto aumento de velocidad de procesamiento, así como la eliminación de ruido y el dar énfasis de características importantes.

La selección de una representación adecuada está relacionada con la selección de una medida de distancia. Muchas representaciones comunes admiten el cálculo aproximado de la distancia euclídea de la serie temporal original pero solo unas pocas admiten la distancia euclídea ponderada. Para series temporales una clasificación [51] se muestra en la Figura 4.19. Hay pocas alternativas para representar series temporales simbólicas.

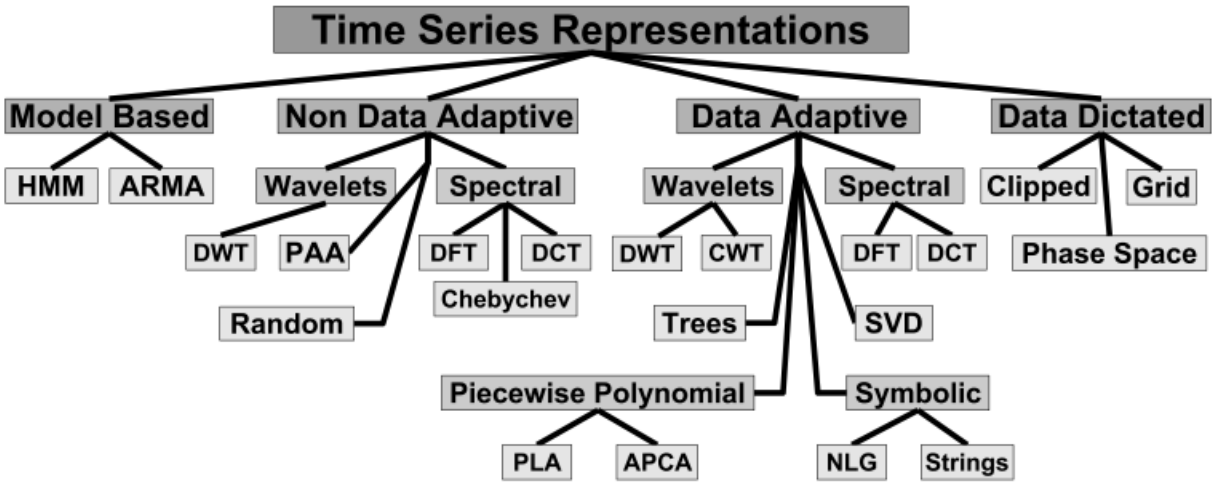


Figura 4.19: Clasificación de los tipos de representación de series temporales.

Al igual que con la mayoría de problemas en minería de datos, la elección adecuada de la representación o aproximación afecta en gran medida a la facilidad y eficiencia de los algoritmos. Debe quedar claro que la utilidad de esta mejora depende en gran medida de la calidad de la aproximación ya que, si la aproximación es muy fiel a los datos originales, entonces la solución obtenida en memoria principal es probable que sea la misma, o muy parecida a la solución que se habría obtenido con los datos originales. Además, el número de accesos a memoria que se harían para confirmar o modificar ligeramente la solución serian insignificantes en comparación con los accesos a disco requeridos de haber trabajado con los datos originales.

Si bien está claro que no hay una representación única que sirva para todas las tareas, la gran cantidad de trabajos sobre minería de datos de series temporales tampoco han aportado ninguna idea de cómo se puede elegir la mejor representación para cada problema.

A continuación, hablaremos sobre algunas de las técnicas de representación de series temporales más conocidas.

4.6.1.1 Piecewise Aggregate Approximation (PAA)

Esta representación reduce los datos de una serie temporal de n dimensiones a N dimensiones dividiéndola en N segmentos del mismo tamaño. El valor medio de cada segmento se calcula y se obtiene como resultado un vector que contiene esos valores medios siendo este vector la representación reducida de la serie original [52]. Cuando $N = n$, la representación transformada es idéntica a la representación inicial. Cuando $N = 1$, la nueva representación es simplemente la media de la secuencia original. De manera más general, la transformación produce una aproximación constante por tramos de la serie original.

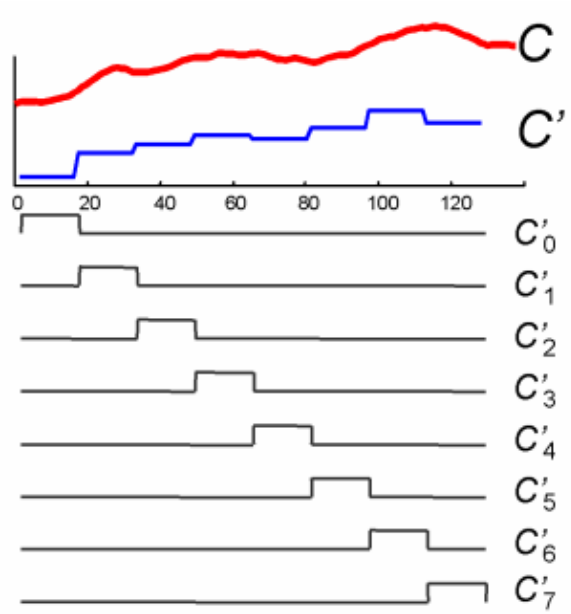


Figura 4.20: Visualización de la técnica de reducción de dimensionalidad PAA

4.6.1.2 Symbolic Aggregate Approximation (SAX)

Es una representación simbólica novedosa para series temporales que se ha demostrado que preserva información significativa de los datos originales y produce buenos resultados en la clasificación y clustering de series temporales.

La idea básica de SAX es la de convertir los datos a un formato discreto usando un alfabeto de tamaño reducido [53]. En este caso, cada parte de la representación contribuye aproximadamente con la misma cantidad de información acerca de la forma de la serie temporal. Para simbolizar una serie temporal hay que normalizarla primero y después llevar a cabo dos pasos de discretización. En primer lugar, una serie temporal T de longitud n se divide en segmentos de igual tamaño w; entonces los valores de cada segmento son aproximados y reemplazados por un único coeficiente que será su promedio. La unión de estos coeficientes w forma la representación a trozos aproximada PAA de T. A continuación, hay que determinar los puntos de corte que dividirán el espacio en α regiones equiprobables, donde α es el tamaño del alfabeto especificado por el usuario. En otras palabras, los puntos de corte se determinan de tal manera que la probabilidad de que un segmento esté en cualquiera de las regiones es aproximadamente la misma. Si los símbolos no son equiprobables, alguna de las subcadenas sería más probable que otra y en consecuencia se introduciría un sesgo en el proceso probabilístico.

Una vez determinados los puntos de corte, se asigna un símbolo a cada región. Los coeficientes de PAA se pueden asignar fácilmente a los símbolos correspondientes a las regiones en las que residen. Los coeficientes se asignan de manera ascendente, es decir, el coeficiente de PAA que está en la región más baja se convierte en "a", la siguiente región por encima de "a" será la "b" y así sucesivamente [54]. En la Figura 4.21 se muestra un ejemplo de cómo una serie temporal se convierte en la cadena $baabccbc$. Nótese que todavía se conserva la forma general de la serie temporal a pesar de la reducción de dimensionalidad, además de equiprobabilidad de los símbolos.

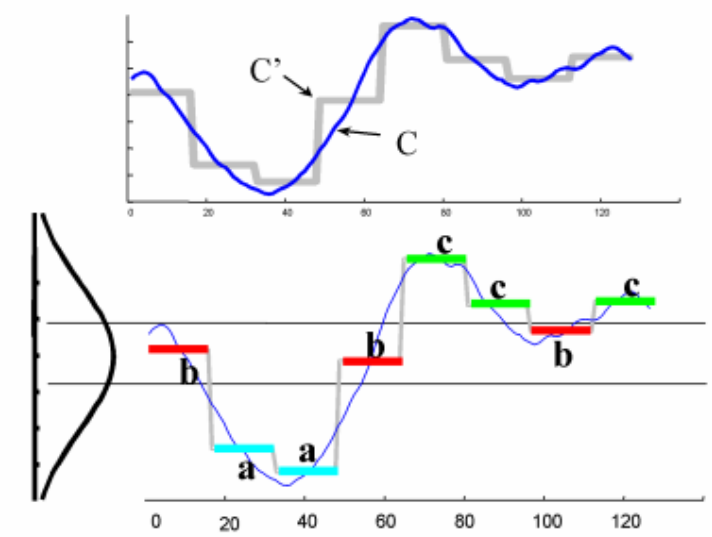


Figura 4.21: Visualización de la técnica de reducción de dimensionalidad SAX.

4.6.2 Visualización

La visualización es una parte importante de la minería de datos porque los seres humanos tienen capacidades notables para detectar patrones ocultos. Sorprendentemente, no ha habido mucha actividad en investigación en la comunidad de minería de datos orientada a la visualización de series temporales. La forma más común es un gráfico de líneas donde el eje horizontal representa el tiempo.

La periodicidad de series temporales se puede visualizar usando espirales [55]. La presentación en espiral puede revelar la estructura periódica mucho mejor que los gráficos lineales. El valor de una serie temporal se puede mostrar en la espiral usando color o grosor en la línea. Espirales entrelazadas pueden representar datos multivaluados.

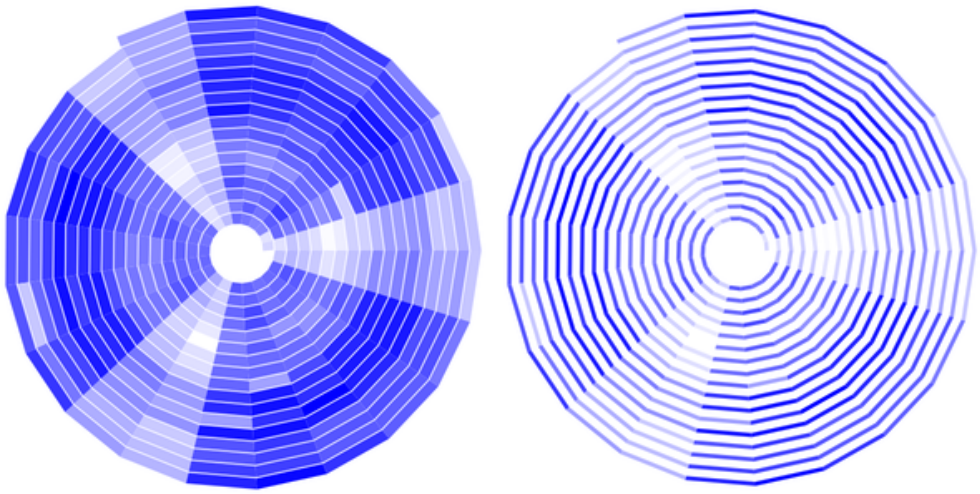


Figura 4.22: Ejemplo de visualización en espiral de una serie periódica.

El VizTree [56] es un árbol que muestra todas las subsecuencias de longitud fija de una serie temporal simbólica obtenida por discretización SAX. Es útil para diferentes tareas de minería de datos de series temporales como detección de anomalías y localización de (sub)secuencias coincidentes. El árbol está vinculado con las representaciones convencionales de series temporales para mostrar los valores originales de los patrones seleccionados por el usuario.

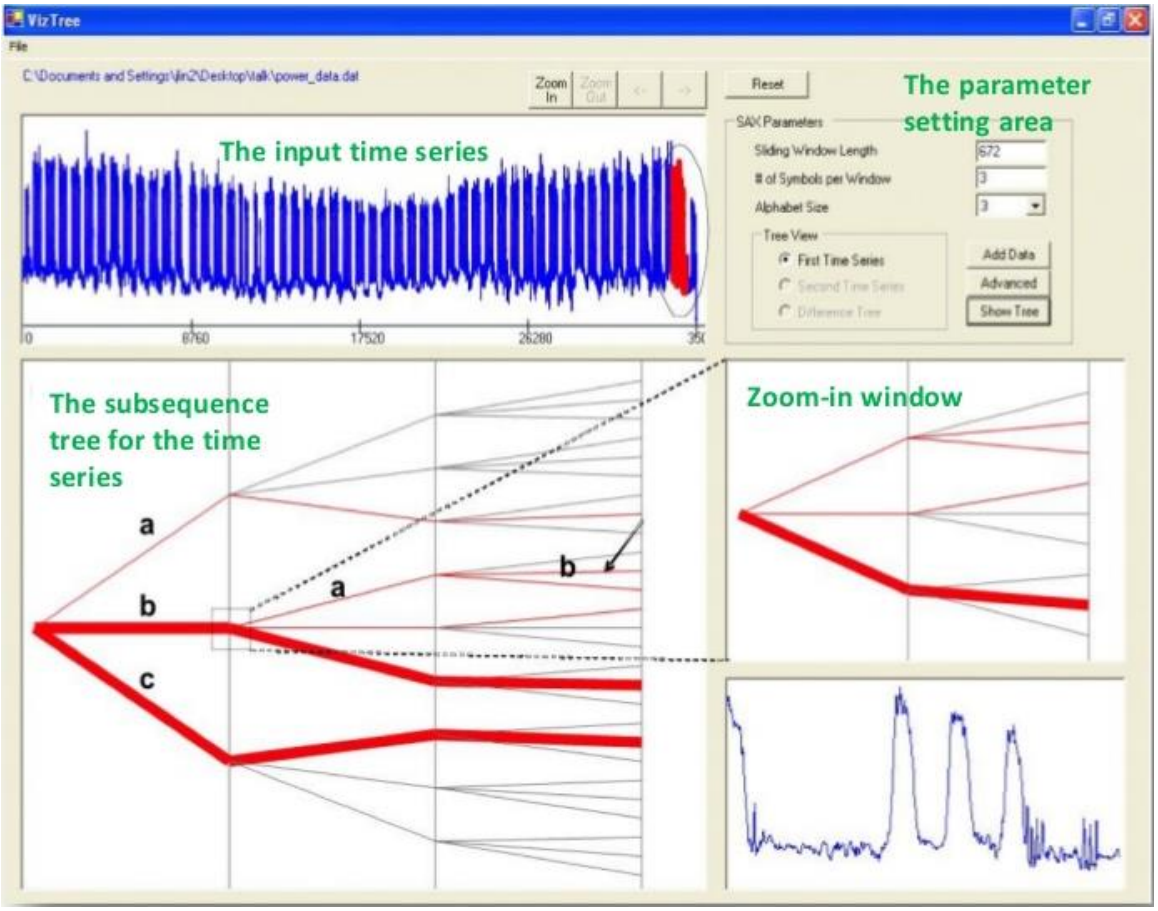


Figura 4.23: Captura de pantalla de VizTree.

Parte IV

Aplicación

5 Descripción de la aplicación

En este apartado vamos a explicar la funcionalidad y la arquitectura de la aplicación. En primer lugar, haremos una pequeña reflexión acerca de porqué elegimos Python como lenguaje para desarrollar la aplicación.

5.1 Porqué usar Python y no R

Python es un lenguaje de programación de propósito general ampliamente utilizado en aplicaciones web por su sintaxis simple y multisistema. Mientras que R es popular por su gran capacidad para la visualización de datos dado que fue desarrollado especialmente para el cálculo estadístico. Ambos lenguajes encabezan la lista de herramientas básicas de cálculo estadístico que un ingeniero dedicado al análisis de datos debe conocer.

Hay un gran debate sobre cuál es más valioso, pero esto depende del punto de vista del programador ya que ambos lenguajes se especializan en ámbitos que pueden complementarse [57].

5.1.1 Análisis de datos

Python ofrece una serie de librerías que ayudan a trabajar con la gran variedad de datos de los que se suele partir. Además, puede facilitar el jugar con los datos permitiendo hacer cualquier cosa que se necesite con ellos, favoreciendo también el escribir código robusto y escalable con un fácil mantenimiento. Pero a diferencia de R, Python no tiene paquetes integrados, aunque tiene soporte para librerías como Scikit, Numpy, Pandas, Sipy y Seaborn, entre otras, muy útiles para tareas estadísticas y de aprendizaje automático. Sin olvidar que la programación en Python es similar al pseudocódigo por lo que se lee y comprende de un vistazo, como si de un lenguaje natural se tratara. No obstante, también permite expresiones matemáticas [58] [59].

R es usado ampliamente para resolver problemas difíciles relacionados con el cálculo estadístico y marketing cuantitativo. Se ha convertido en una herramienta esencial para el análisis de finanzas y negocios en compañías como Google o Twitter.

R es un lenguaje de código abierto que puede usarse en cualquier plataforma de sobremesa. Tiene un innovador sistema de paquetes que permite ampliar la funcionalidad y puede ser fácilmente integrado con otros lenguajes de programación con orientación a objetos tales como C, C++ y Java. Su sintaxis es simple e intuitiva con lo que cualquier profesional con conocimientos mínimos de programación puede manejarlo sin problema. Además, R es el lenguaje más utilizado por la comunidad investigadora en análisis de datos para diseminar las técnicas estadísticas y de minería de datos desarrolladas.

5.1.2 ¿Qué convierte a Python en el más extendido?

En Python todo son objetos por lo que es posible crear aplicaciones usando varios paradigmas de programación de una forma clara y comprensible.

- **Amplitud:** Python dispone de un índice de paquetes público llamado PyPi con multitud de add-ons bien organizados que permiten al programador ahorrar tiempo de desarrollo. Por lo tanto, si un programador quiere hacer algo con Python es muy probable que ya haya algo

implementado con lo se evita empezar de cero.

- **Eficiencia:** Hoy en día se gasta mucho tiempo en definir y procesar grandes volúmenes de datos. Con la creciente cantidad de datos que necesitan ser procesados se vuelve extremadamente importante el poder gestionar de manera eficiente el uso de la memoria. Python tiene la capacidad de procesar los datos de forma iterativa cogiendo un dato a la vez y recorriendo toda una cadena de procesamiento. Además ofrece una herramienta de migración llamada *collective.transmogrifier* que ayuda a realizar actualizaciones complejas e interdependientes de los datos a medida que se van procesando. Esta herramienta juega un papel vital cuando se trata de grandes conjuntos de datos.

- **Gran dominio del lenguaje:** La popularidad de Python se debe en gran parte a una sintaxis clara y fácil aprendizaje guiado.

5.1.3 ¿Por qué usar R para análisis de datos?

R es una de las mejores herramientas para análisis de datos en el mundo de la visualización de datos. Tiene prácticamente todo lo necesario para trabajar con datos. Se pueden crear visualizaciones únicas y muy variadas que van más allá de simples tablas o gráficos de barras. Está diseñado especialmente para el análisis de datos y tiene una flexibilidad para mezclar y combinar diferentes modelos estadísticos y predictivos para obtener los mejores resultados posibles. Una gran característica de R es la posibilidad de reproducir un trabajo a partir de los datos originales y conseguir los mismos resultados. Por último, R tiene una gran comunidad detrás que no hacen sino aumentar las posibilidades del lenguaje con nuevas funcionalidades en forma de paquetes.

5.1.4 Limitaciones

Python es un lenguaje interpretado y por lo tanto es más lento que un lenguaje compilado, sin embargo, muchos de sus paquetes han sido optimizados en los últimos años llegando a una velocidad similar a la de C. Python además es un lenguaje de tipos polimórficos que plantea ciertas restricciones de diseño y requiere de pruebas rigurosas ya que los errores aparecen solo en tiempo de ejecución.

La curva de aprendizaje de R no es trivial para personas que no tienen una base en programación o provienen del mundo de interfaces gráficas. Trabajar con R puede ser muy lento si el código está mal escrito, aunque tiene soluciones en forma de paquetes para ayudan a paliar el problema.

5.1.5 Conclusiones

Cuando empezamos a plantear este proyecto queríamos, por una parte, que nuestras aplicaciones pudieran ser potencialmente utilizadas, ampliadas o mejoradas por el mayor número de personas posibles. Por este enfoque que queremos dar al proyecto hemos decidido trabajar con Python dada su versatilidad y facilidad para trabajar con ficheros de datos.

Además, creemos que aprender a utilizarlo nos será útil en el futuro.

5.2 Objetivo de la aplicación

Como ya se ha comentado en el apartado del software del fabricante, SenseWear es una aplicación bastante completa que permite mucho detalle en la visualización de los datos registrados por el brazalete. Sin embargo, tiene una finalidad muy abierta y no se centra en ninguna tarea de análisis concreta como podría ser encontrar patrones de sueño del individuo o su estilo de vida en cuanto a ejercicio físico, por ejemplo. Podría decirse que está enfocada a la representación gráfica de todos los datos del fichero y a proporcionar un informe resumen.

Si bien es posible observar cada episodio y cada instante deseado, puede resultar tedioso y llevar demasiado tiempo por la gran cantidad de interacción necesaria para recopilar toda la información requerida. Por ejemplo, para comparar dos episodios de sueño de días distintos, habría que seleccionar el día deseado, acotar el inicio y final del instante a visualizar desplazándose a izquierda o a derecha en el eje de abscisas que corresponde al tiempo, después habría que ajustar los filtros para visualizar sólo los parámetros necesarios y elegir para cada uno las unidades de representación y repetir el mismo proceso para el segundo episodio con el que se desea comprar, perdiendo el foco del primero. Es por ello que decidimos, mientras íbamos analizando los datos que teníamos, realizar una aplicación que no sólo nos facilitara esta tarea, sino que pudiera ser de utilidad médica por ofrecer una visualización rápida y sencilla de la información recopilada.

Nuestra aplicación aborda la visualización general de todos los episodios, el análisis de los episodios de sueño y el análisis del consumo energético. Para ello, primero selecciona los instantes de comienzo y final de cada episodio de sueño y de actividad física, incluidos los sedentarios para posteriormente representarlos en distintos formatos dependiendo de la información que se pretende mostrar en cada pestaña. El análisis de sueño engloba la visualización de cada episodio de sueño en detalle, con parámetros como la temperatura y el flujo térmico, agrupa los episodios de sueño mediante técnicas de minería de datos como el clustering para observar patrones en la rutina de sueño del paciente. Por otro lado, el análisis de consumo energético presenta de un vistazo un resumen con las calorías consumidas en cada tipo de actividad física, por orden cronológico y por día.

Por lo tanto, el objetivo de la aplicación es facilitar el análisis del sueño y la actividad física de un paciente que ha llevado colocado el brazalete un periodo de tiempo, centrándose para ello en los distintos tipos de episodios de sueño y de actividades.

5.3 Descripción de la funcionalidad

La aplicación consiste en una única ventana con varias pestañas donde cada una proporciona distinta información. Una vez se ejecuta, se muestra una pantalla vacía con todas las pestañas. Para empezar a visualizar la información hay que pulsar el botón de cargar ficheros, tal como se muestra en la Figura 5.1.

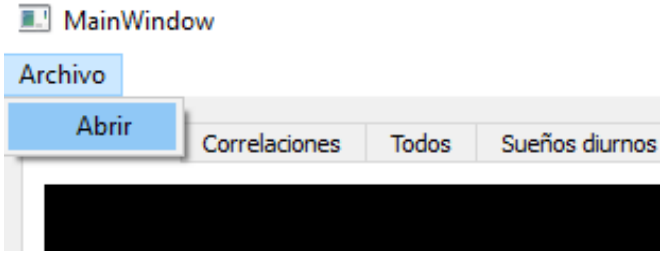


Figura 5.1: Selección de fichero a cargar

La información se divide en episodios y cada uno tienen asignado un nombre identificativo para localizarlos entre las diferentes pestañas. La nomenclatura que se sigue es la siguiente:

$$fecha - TipoNum$$

- *Fecha* es el día en el que se produce el episodio.
- *Tipo* marca los distintos tipos de episodios, que pueden ser:
  - Su para episodios de sueño.
  - Se para episodios sedentarios.
  - Li para episodios de actividad ligera.
  - Mo para episodios de actividad moderada a intensa.
- Num es el número de episodio respecto a su tipo, por ejemplo, $Se4$ se corresponde con el cuarto episodio sedentario.

Se puede ver un ejemplo en la Figura 5.2 donde se aprecia que está seleccionado el primer episodio de sueño del día 15.

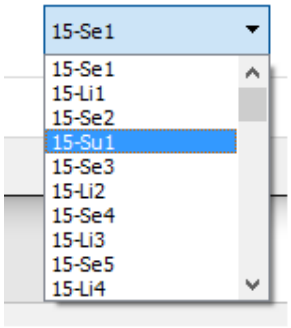


Figura 5.2: Nombres de episodios

A continuación, se describen los colores usados en la aplicación:

- Azul: en la barra de actividades representa el sueño, se usan varias tonalidades siendo el más claro un sueño ligero y el más oscuro un sueño muy profundo. Cuando se usa en gráficas de series temporales representa la temperatura
- Amarillo: actividad sedentaria
- Naranja: actividad ligera
- Rojo: actividad moderada
- Verde: flujo térmico

5.3.1 Pestaña 1: Visualización de episodios de sueño

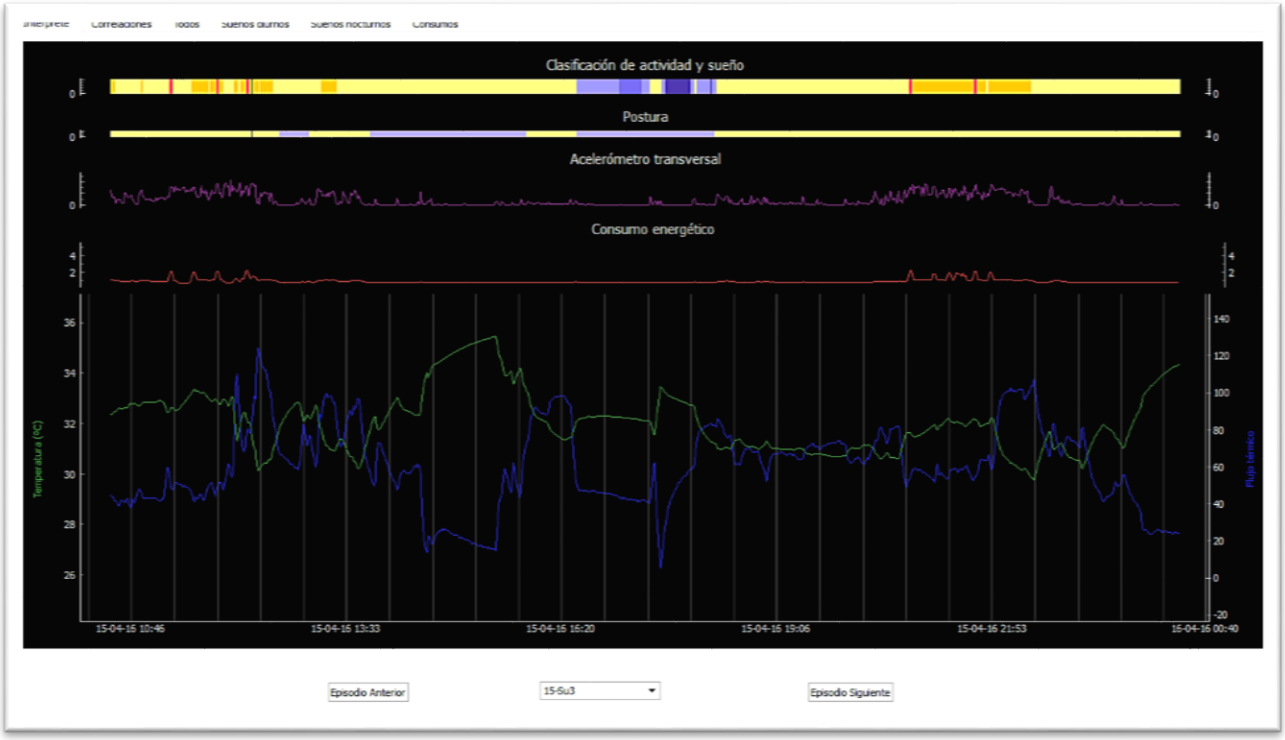


Figura 5.3: Pestaña del intérprete de sueños de la aplicación

La primera pestaña permite visualizar en detalle cada episodio de sueño. Un episodio de sueño es un intervalo de tiempo que empieza en el instante en que el paciente se duerme hasta el momento en que se despierta y permanece así más de 30 minutos seguidos. De esta forma se evita dividir un mismo sueño en varios episodios distintos, ya que es habitual que el paciente se despierte en varias ocasiones mientras está durmiendo. Por lo tanto, se podrán apreciar todas las interrupciones que se produzcan, mostrándose despierto.

En esta pestaña cada episodio a visualizar está contenido en un intervalo que contiene las 6 horas anteriores al inicio del sueño y las 6 horas posteriores a su finalización. Así se puede contemplar la actividad del paciente antes y después de conciliar el sueño para intentar averiguar el motivo de posibles trastornos del sueño. Por ejemplo, es posible que, si el paciente realiza ejercicio intenso antes de dormir no pueda conciliar el sueño rápidamente o no tenga un sueño lo suficientemente profundo y por tanto la calidad del sueño sería menor que si realiza ejercicio en otro momento del día.

Todos los ejes de las gráficas tienen la misma escala y ésta es fija para todos los episodios del mismo paciente. Es decir, en todos los episodios de dicho paciente las gráficas se ajustan siempre a un rango de valores determinado, que se calculan como el registro mínimo y el registro máximo de cada parámetro. De esta forma se pueden comparar episodios distintos de forma más efectiva que si las escalas fueran variables, lo que daría visualizaciones distorsionadas y difícilmente comparables.

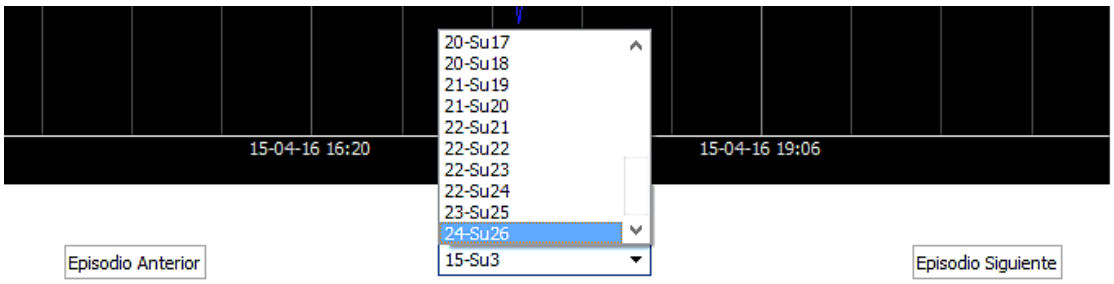


Figura 5.4 Selección de episodios de sueño

Esta pestaña también dispone de un selector de episodio desplegable y dos botones para avanzar y retroceder entre episodios por orden cronológico. En este caso y según la Figura 5.4 se puede observar que hay un total de 26 episodios de sueño repartidos entre los 9 días de duración de la monitorización.

En el panel principal de esta pestaña se muestra una gráfica donde se representan varios parámetros a la vez como son la clasificación de actividad física y tipos de sueño. Otras gráficas muestran la postura del paciente, las medidas registradas por el acelerómetro transversal, el consumo energético, la temperatura y el flujo térmico. Todos los parámetros comparten el mismo eje de abscisas, donde se muestra la fecha y hora, minuto a minuto de cada parámetro registrado por el dispositivo.

La primera gráfica de barras indica con un color el tipo de actividad física o el tipo de sueño en el que se encuentra el paciente en cada minuto. Mediante la variación de colores se puede apreciar una mayor intensidad de actividad física o profundidad de sueño según el tono del color. Los colores azules representan el sueño y el resto corresponden con la actividad física, siendo el amarillo actividad sedentaria y el rojo actividad moderada.

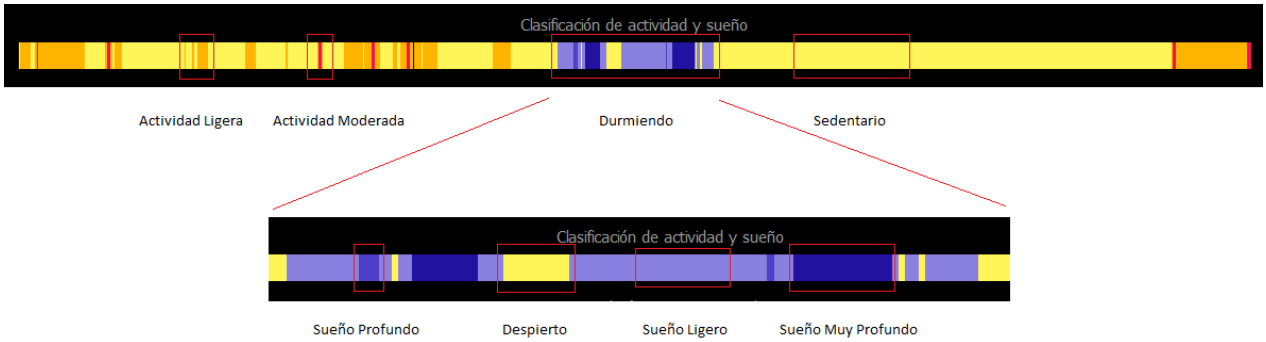


Figura 5.5 Tipos de episodios en la gráfica de barras

En el caso de la Figura 5.5, se trata de un episodio de casi dos horas de duración que se produce durante el día. Se puede apreciar cómo el paciente ha realizado varios tipos de actividad física antes de dormir, aunque de poca intensidad ya que la duración de los episodios moderados es de unos pocos minutos. Haciendo zum en el episodio también se puede apreciar que el paciente no tiene un sueño continuado, pues se despierta hasta en cuatro ocasiones antes de despertarse finalmente. Además, llega a alcanzar el sueño muy profundo en poco tiempo y en varias ocasiones, lo cual puede ser un indicativo de que el paciente no duerme bien por las noches.

A continuación, se encuentra una segunda gráfica de barras que indica la postura del paciente. Si está tumbado se marca con color azul y si no lo está se marca con color amarillo. Esta gráfica permite conocer el tiempo que tarda el paciente en dormirse tras haberse acostado lo que también puede ser un indicativo de su eficacia del sueño. Junto con la curva del acelerómetro también se puede observar si se muestra inquieto al acostarse o si en las interrupciones del sueño se ha levantado o

permanece acostado por nombrar algunas utilidades.

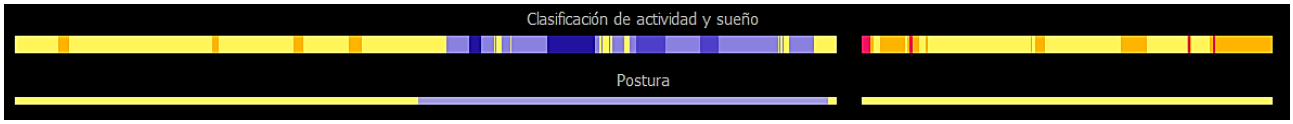


Figura 5.6: Gráfica de postura

En la Figura 5.6 aparece otro episodio de sueño donde se aprecia que el paciente se acuesta y tarda unos minutos (20 aproximadamente) en dormir. Se producen algunas interrupciones en el sueño, aunque sigue acostado hasta que despierta finalmente y al rato se levanta. Tras levantarse se produce una corta interrupción en los registros del dispositivo lo que indica que el paciente se lo ha retirado.

Las dos primeras curvas de la gráfica representan el movimiento del paciente detectado por el acelerómetro anterógrado del dispositivo y el consumo en cada instante. Conocer la cantidad de movimiento resulta útil para saber si el paciente se mueve durante el sueño y tener indicios de posibles patologías relacionadas con el movimiento, como apneas del sueño o el síndrome de piernas inquietas. También permite saber si el paciente lleva un estilo de vida sedentario. El consumo junto con el movimiento permite dar a conocer el consumo basal del paciente en momentos de sedentarismo y las calorías consumidas en momentos de actividad física para saber si el ejercicio que realiza es suficiente y tener un nivel de detalle mayor al que ofrece el gráfico de barras de colores. También es una forma de discernir entre ejercicio físico, que consumirá más calorías en un tiempo menor o simplemente movimiento. El consumo viene dado en kilo-calorías.

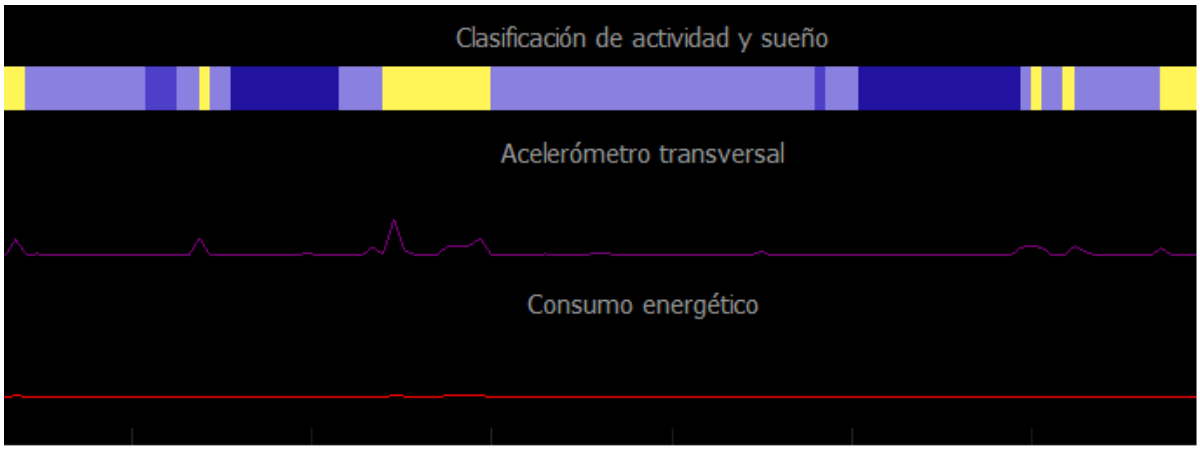
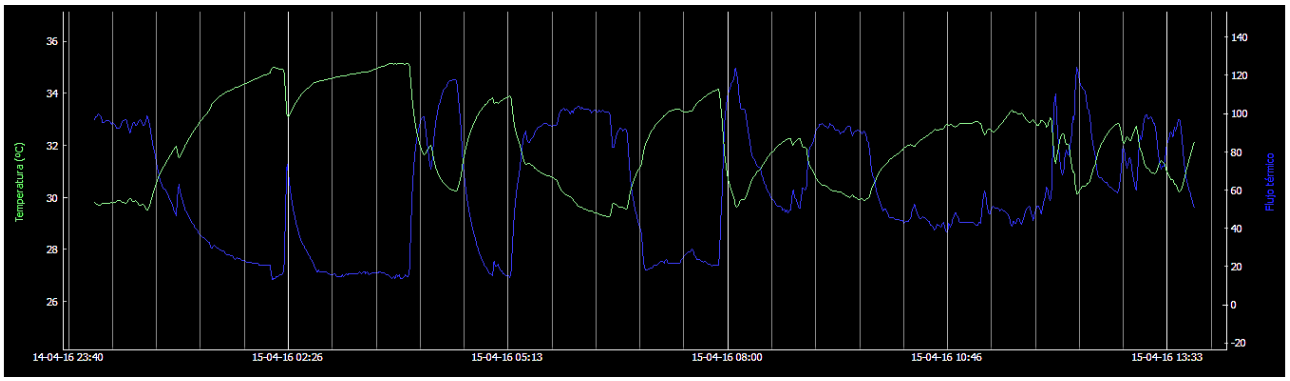


Figura 5.7 Curvas de movimiento y consumo

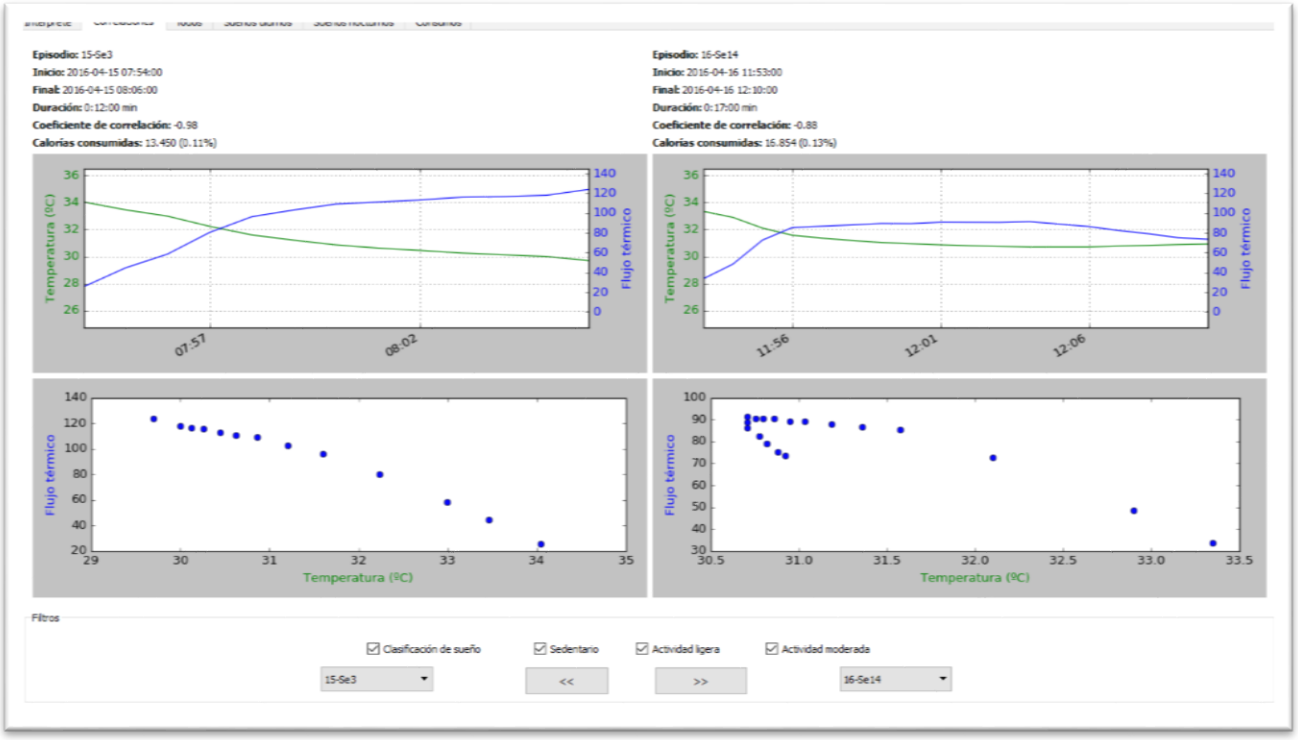
En la Figura 5.7 se puede apreciar que durante el sueño el paciente prácticamente no se mueve salvo cuando se despierta, por lo que en principio no hay indicativos de problemas.



Las dos últimas series temporales se corresponden con la temperatura y el flujo térmico. Estos parámetros son los que se usarán posteriormente para relacionar y agrupar episodios de sueño. Esta

pestaña permite una mejor visualización y una mayor amplitud del episodio. Como se puede apreciar en la figura, la curva de la temperatura y la curva del flujo insinúan un comportamiento inverso, de forma que cuando aumenta la temperatura disminuye el flujo térmico y viceversa. En la fase del sueño lo habitual es que la temperatura tienda a aumentar unos pocos grados y por tanto el flujo térmico disminuya.

5.3.2 Pestaña 2: Relaciones entre temperaturas durante el sueño



Mientras que en la pestaña anterior solo se filtraban episodios de sueño y se mostraba cada uno en un amplio intervalo de tiempo, en esta pestaña se pretende desglosar los datos en distintos tipos de episodios, que pueden ser de sueño y de actividad física. Estos últimos a su vez se componen de episodios sedentarios, de actividad ligera, moderada e intensa. La finalidad es representar más información de cada uno de estos episodios y observar cómo efectivamente hay una correlación entre la temperatura y el flujo térmico en cada episodio.

Para ello, lo primero que hace es agrupar los datos según el tipo de episodio que sea. Un episodio debe cumplir las siguientes consideraciones:

- Su duración debe tener un mínimo de 5 minutos.
- Permite pequeñas interrupciones intermedias de un máximo de 35 minutos para episodios de sueño, de 7, 4 y 3 minutos para episodios de actividad sedentaria, ligera y moderada respectivamente. Se consideran interrupciones tanto la retirada del brazalete como estar en otro tipo de episodio.
- El episodio se considera finalizado cuando se produce una interrupción mayor a la máxima permitida. Por tanto, el minuto de finalización del episodio será el último correspondiente al tipo actual.

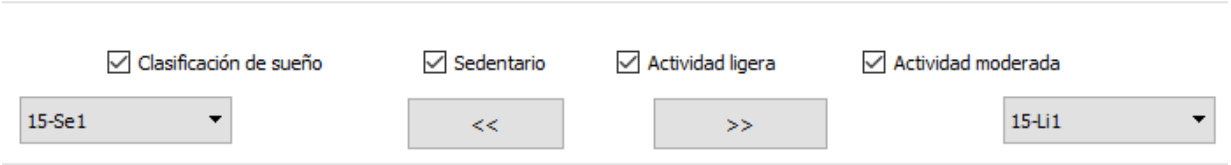


Figura 5.8 Filtro y selector de episodios

Una vez clasificados los episodios por tipo, se muestran dos episodios al mismo tiempo, uno

en la columna izquierda y otro en la derecha, para poder compararlos. Los episodios a mostrar se seleccionan con sus respectivos selectores desplegables o bien desplazándose entre ellos cronológicamente mediante los botones de avanzar y retroceder, de igual forma que en la primera pestaña. El desplazamiento es de uno en uno, por lo que al avanzar la columna izquierda mostrará el episodio de la derecha y la columna derecha mostrará el episodio siguiente. También se pueden filtrar los episodios seleccionando los que se quieran visualizar en las correspondientes casillas de activación, tal como se muestra en la Figura 5.8 en la que todos los tipos de episodio están activados.

Episodio: 15-Se1
Inicio: 2016-04-15 00:00:00
Final: 2016-04-15 00:31:00
Duración: 0:31:00 min
Coeficiente de correlación: -0.93
Calorías consumidas: 28.439 (0.23%)

Figura 5.9 Información del episodio visualizado

Tal como se aprecia en la Figura 5.9, por cada episodio se proporciona información útil sobre sus características. Así, se muestra el nombre y tipo, la fecha completa de inicio y fin, la duración, el coeficiente de correlación entre la temperatura y el flujo térmico y el total y proporción de calorías consumidas. El coeficiente de correlación r varía en el intervalo $[-1,1]$ y se interpreta de la siguiente forma:

- Si $r = 1$, existe una correlación positiva perfecta. El coeficiente indica una dependencia total entre las dos variables llamada relación directa: cuando una de ellas aumenta, la otra también lo hace en proporción constante.
- Si $0 < r < 1$, existe una correlación positiva.
- Si $r = 0$, no existe relación lineal.
- Si $-1 < r < 0$, existe una correlación negativa.
- Si $r = -1$, existe una correlación negativa perfecta. El coeficiente indica relación inversa lo que quiere decir que cuando una de ellas aumenta, la otra disminuye en proporción constante.

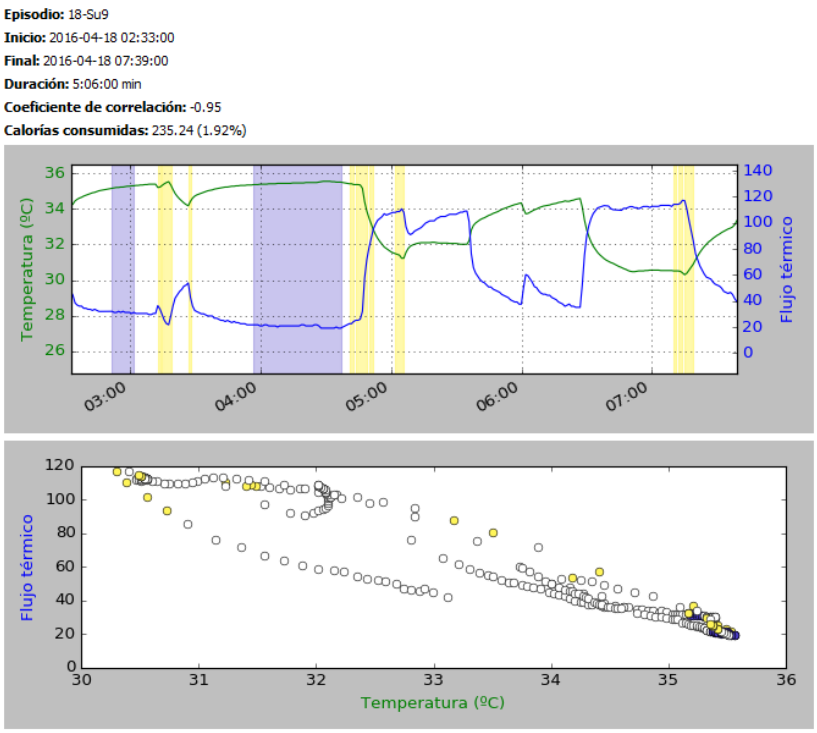


Figura 5.10: Ejemplo de interpretación de coeficiente de correlación.

En el ejemplo de la Figura 5.10 el coeficiente de correlación es de -0.95 lo que quiere decir que existe una correlación negativa y el estar cercana a -1 nos dice también que cuando la temperatura aumenta el flujo térmico disminuye o la inversa. Este hecho se puede comprobar en la gráfica superior donde se ven las dos series temporales del flujo térmico y la temperatura que se acercan o se alejan en el mismo instante de tiempo, lo que indica que cuando una de ellas aumenta, la otra disminuye y a la inversa. También nos da buena idea de ello la forma del diagrama de dispersión que se asemeja a una recta decreciente.

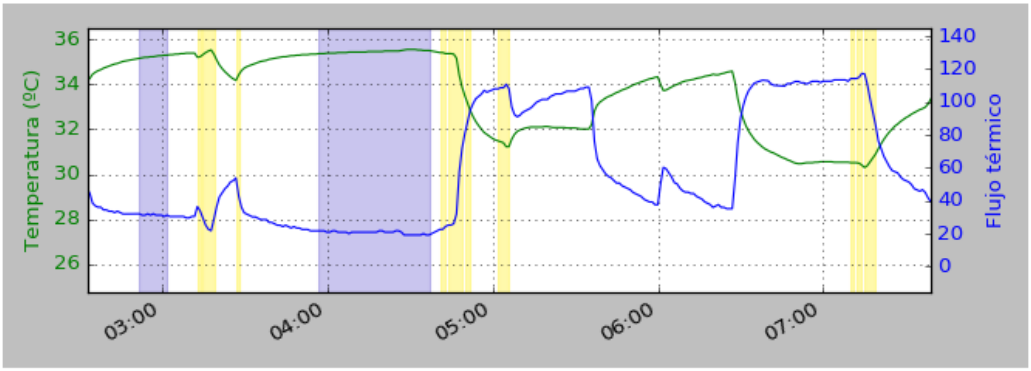


Figura 5.11: Captura de gráficas de flujo temporal y temperatura

Además, tal como se aprecia en la Figura 5.11 las franjas de colores del fondo indican la aparición de una interrupción en el episodio por cambiar el estado actual del paciente. Por ejemplo, en la captura el paciente está dormido y sin embargo hacia las 3:25 se despierta, y tras unos minutos vuelve a dormirse. Las franjas de color azul son los minutos de sueño profundo y muy profundo. El código de colores se corresponde con el descrito en el apartado 5.1.

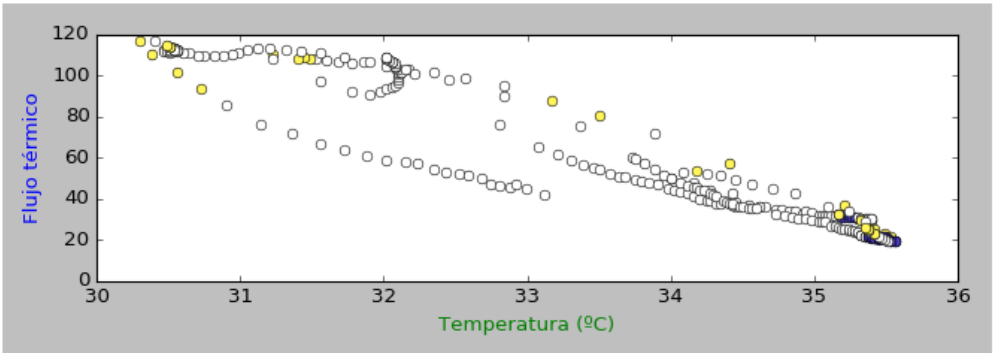


Figura 5.12: Diagrama de dispersión

En el gráfico de dispersión también se colorean los puntos según el tipo de sueño para los episodios de sueño.

5.3.3 Pestañas 3 a 5: Clustering de sueños

Figura 5.13 Pestaña de clustering

Dada la relación observada entre temperatura y flujo térmico, se puede utilizar uno de los métodos anteriormente mencionados para clasificar los episodios de sueño. De esta forma podemos encontrar grupos de episodios similares, grupos diferentes y episodios muy diferentes al resto. Esta clasificación se puede hacer respecto a la temperatura y flujo térmico por un lado y respecto al consumo energético por otro, lo que dará lugar a distintas agrupaciones. Además, es posible seleccionar los episodios a agrupar para ignorar por ejemplo posibles episodios anómalos.

La técnica de clustering utilizada es clustering jerárquico aglomerativo, que ofrece un dendrograma, una estructura de agrupación completa, y computacionalmente es más eficiente que el divisivo.

Para realizar esta técnica primero hay que normalizar los datos para que las distancias a calcular no queden distorsionadas debido a la diferencia de rangos entre los valores de las series a comparar. Por ejemplo, al comparar dos series temporales de temperatura, una con un rango de entre 28º a 36º y otra con otro rango de 34º a 36º, obtendremos distancias muy dispares para pequeñas variaciones de temperatura. Al normalizar ambas series, se perdería la diferencia de rangos y por lo tanto las variaciones de temperatura quedarían disimuladas, consiguiendo una comparación mucho más efectiva. Normalizando obtenemos los valores con media cero y varianza unidad con los que poder trabajar.

El siguiente paso consiste en calcular las distancias entre las series temporales, es decir, entre los episodios. En el caso de las agrupaciones entre temperatura y flujo, las distancias se calculan por separado. Por un lado, se calculan las distancias de temperatura entre cada episodio y por otro lado el flujo. En el caso de las agrupaciones por consumos, simplemente se calculan las distancias de consumo energético entre los episodios. Las distancias se calculan por Dynamic Time Warping (DTW) que permite una medida más adecuada para series temporales que la distancia eucídea ya que

no se ve afectada por la alineación temporal de las series a comparar. Así, dos series temporales normalizadas que tengan una forma muy similar, obtendrán distancias cortas, a pesar de que puedan estar desplazadas entre sí. En el caso de las agrupaciones de temperatura y flujo el siguiente paso es calcular la distancia total entre ambas distancias. Para ello utilizamos la distancia euclídea con las distancias previamente calculadas.

Los resultados de calcular las distancias los almacenamos en una matriz para proceder finalmente al cálculo de los agrupamientos mediante clustering jerárquico aglomerativo. La técnica de encadenamiento utilizada es agrupamiento de promedios mediante el algoritmo UPGMA que construye un dendrograma que refleja la estructura de los grupos según su similitud. La matriz de distancias queda representada en forma de tabla como se muestra en la Figura 5.16 y los resultados del clustering en el dendrograma de la Figura 5.17.

Esta pestaña sirve para buscar los episodios menos comunes y también para poder compararlos con otros. En la Figura 5.13 se ve la división en dos zonas donde cada una nos da distinta información, en la parte izquierda se encuentra el selector del tipo de clustering, la matriz de distancias y el dendrograma generados tras aplicar clustering o agrupamiento para clasificar los episodios de sueño del paciente y la parte derecha se dedica a la comparación manual de episodios.

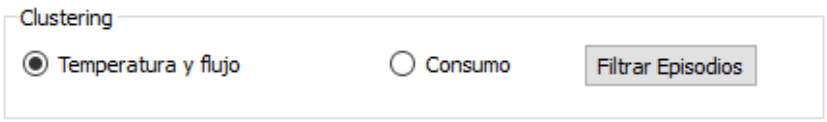


Figura 5.14: Selector de tipo de clustering

El selector de la Figura 5.14 permite elegir las variables a comparar de cada episodio, flujo térmico y temperatura o el consumo energético. Inicialmente el clustering se realiza sobre todos los episodios, aunque se pueden filtrar con el botón correspondiente eligiendo los que se quieran utilizar para el agrupamiento como se ve en la Figura 5.15.

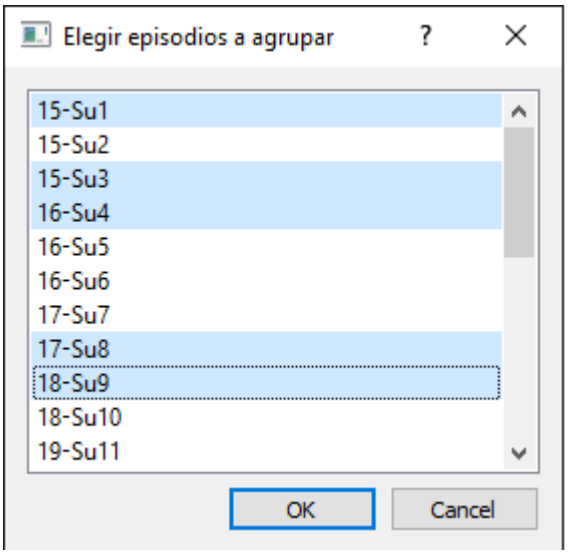


Figura 5.15: Filtro de episodios de sueño.

La matriz de distancias de la Figura 5.16 muestra la distancia entre cada par de episodios. No hay diferencia en la dirección, es decir, la distancia entre el episodio 1 y el episodio 2 es la misma que entre el episodio 2 y el 1. Y a menor distancia, mayor es la similitud entre dos episodios. Se representa en una tabla como un mapa de calor donde en verde se muestran las distancias más cercanas y en naranja las más lejanas, pasando por una distancia intermedia en amarillo.

Matriz de distancias entre episodios de sueño

	15-Su2	15-Su3	16-Su5	16-Su6	18-Su10	19-Su14	20-Su18	21-Su20	22-
15-Su3	115.2								
16-Su5	35.5	117.0							
16-Su6	90.8	64.7	49.8						
18-Su10	35.0	121.7	3.2	53.9					
19-Su14	56.4	135.6	47.8	100.3	47.6				
20-Su18	96.2	72.4	97.3	71.5	100.9	56.9			
21-Su20	36.8	109.2	4.7	40.9	6.3	47.2	96.3		
22-Su22	32.8	36.8	32.5	24.9	33.8	76.5	47.6	32.2	
22-Su23	90.4	81.8	118.3	121.3	117.8	114.5	93.1	112.6	84.9
22-Su24	35.7	105.4	3.6	38.0	4.4	46.2	91.7	2.1	31.2

Figura 5.16: Matriz de distancias

El dendrograma de la Figura 5.17 sirve para explicar la forma de interpretarlo. Este dendrograma es el resultado del agrupamiento realizado donde se puede ver en forma de árbol las categorías que van formando los episodios dependiendo de la distancia obtenida entre unos y otros.

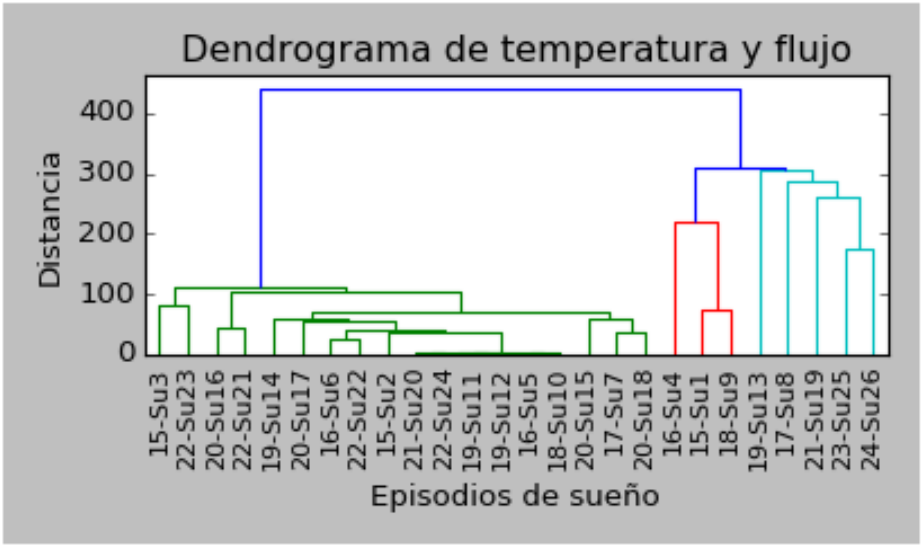


Figura 5.17: Captura de dendrograma

Este agrupamiento ha concluido con tres grandes grupos (verde, rojo y turquesa) que a su vez se dividen en otros más pequeños. El grupo verde se compone de muchos episodios bastante parecidos entre sí. Sin embargo, el grupo rojo y turquesa, menos numerosos, tienen episodios más dispares. Además, se observa que el grupo verde se parece menos al rojo y menos aún a los episodios del grupo azul, dada la distancia que hay entre ellos.

La distancia se representa en el eje de ordenadas lo que significa que cuanto menor sea más parecidos serán los episodios, es decir cuanto más bajo se encuentre una línea horizontal uniendo a dos grupos, más similares son entre sí. Se puede ver un ejemplo en la Figura 5.18 donde los episodios $21-Su20$ a $18-Su10$ son los que más se asemejan seguidos de $16-Su6$ y $22-Su22$. Siguiendo con la explicación, el grupo formado por $21-Su20$ a $18-Su10$ a su vez forma un grupo

con $15 - Su2$ al existir una línea horizontal que los une. El siguiente grupo se crearía con el que acabamos de comentar y por el formado por $16 - Su6$ y $22 - Su22$. Este proceso se repite hasta llegar al grupo final formado por todos los episodios y que estaría en lo más alto del eje de ordenadas.

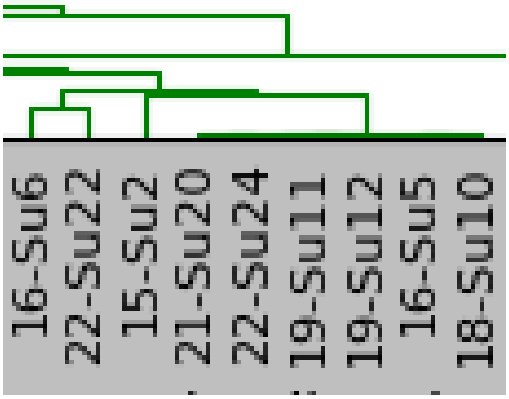


Figura 5.18: Ejemplo de grupos similares en dendrograma

Esto se hace para intentar encontrar inicialmente episodios de sueño que puedan ser anómalos, ya que destacarían por estar a gran distancia de otros episodios de sueño.

En este caso, tras comprobar el dendrograma y las pestañas correspondientes a los sueños diurnos y los nocturnos, encontramos que la mayoría de los episodios pertenecientes al grupo rojo y al turquesa forman parte de sueños nocturnos, mientras que los del grupo verde son en su mayoría sueños diurnos. Sin embargo, el episodio $16 - Su4$ perteneciente al grupo rojo es un sueño diurno muy diferente al resto de sueños del grupo verde, lo puede indicar una anomalía que puede observarse en las correspondientes gráficas.

Se tiene la opción de comparar la forma de las series temporales de dos episodios de sueño en la parte derecha de la pestaña pudiendo elegir el episodio deseado con el selector de la parte superior. De esta forma se puede observar la forma y detalles de los episodios de cada grupo.

En la Figura 5.19 se muestran dos episodios completos, cada uno con la fecha y la hora a la que transcurre. Como en la pestaña anterior, aquí también se dibujan las gráficas de temperatura y flujo térmico, además del consumo energético, y se usan franjas verticales para destacar el momento en que el paciente se despierta (en amarillo) o los minutos de sueño profundo o muy profundo (en azul). Se puede ver una comparación entre los episodios $16 - Su6$ y $22 - Su22$ donde queda claro visualmente que las formas de las series temporales de temperatura y flujo térmico son muy similares.

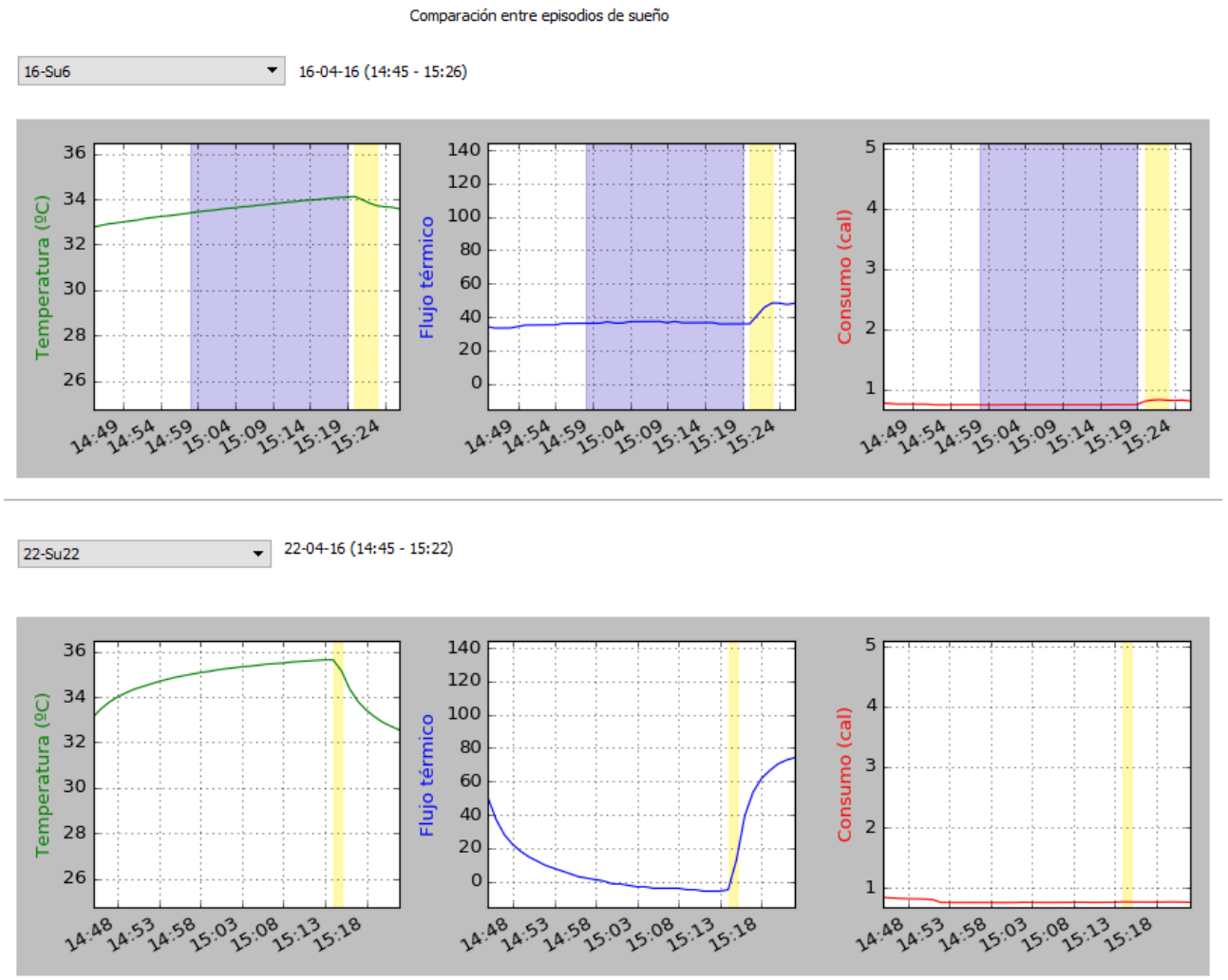


Figura 5.19: Comparación de dos episodios de sueño

En las pestañas 4 y 5 se separan los episodios de sueño diurno de los nocturnos para realizar clustering de forma independiente. Es importante separar los episodios diurnos de los nocturnos ya que, debido a la diferencia horaria la duración también es muy diferente. Al hacer clustering de todos los episodios debería verse esta división y de ser así no se aporta información relevante. Debido a esto es que hemos separado los sueños para analizarlos.

Por lo tanto, estas dos pestañas seguirán el mismo formato que la tercera solo que analizando los episodios diurnos y nocturnos por separado.

5.3.4 Pestaña 6: Consumos

Figura 5.20: Pestaña consumos de la aplicación

Tras analizar los episodios de sueño, el segundo enfoque de la aplicación consiste en el análisis de la actividad física. En esta pestaña se pretende mostrar de un vistazo el estilo de vida que lleva el paciente cuando no está durmiendo, en cuanto a ejercicio y consumo calórico se refiere. Para lo cual se muestran varias aproximaciones desde distintos puntos de vista empleando cuatro tipos de gráficas.

En este caso los datos del paciente se dividen en días completos de 24 horas comenzando a las 00:00h de la noche hasta las 23:59h del mismo día. Los distintos tipos de episodios contenidos en cada día se obtienen de la misma manera que en los apartados anteriores, aunque pueden no coincidir en número debido a que los episodios que queden entre dos días, serán contados como dos episodios distintos. Por lo tanto, la única forma de ver en detalle un episodio concreto será mediante su tipo y número de día correspondientes.

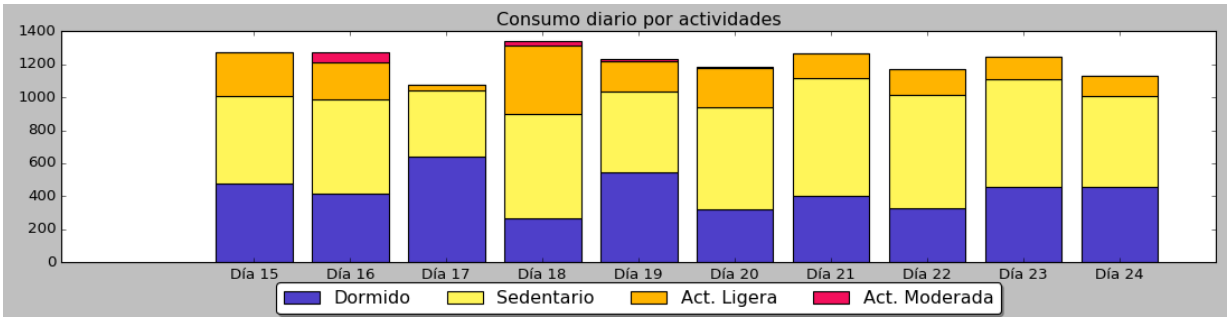


Figura 5.21: Gráfico de barras

Lo primero que se ve es la gráfica de barras de la Figura 5.21, donde las barras representan los días registrados por el brazalete. Cada barra se divide en colores con el tipo de actividad realizado cuya altura es la suma de las calorías consumidas. Esta gráfica sirve para hacerse una idea del consumo calórico del paciente día a día.

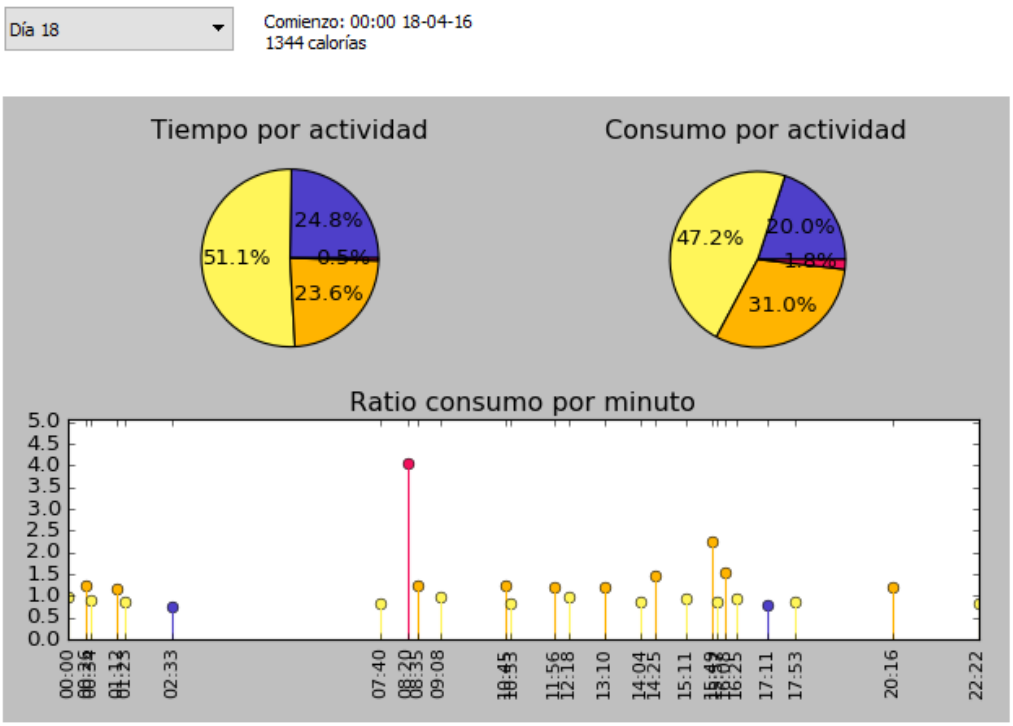


Figura 5.22: Gráficas de consumo correspondientes a un día

La segunda parte de la ventana corresponde con la comparación de dos días elegidos mediante los selectores desplegables. La información de un día se desglosa en tres gráficas. La primera muestra el porcentaje de tiempo total invertido en cada tipo de episodio. La segunda muestra el porcentaje de calorías consumidas en el día por actividad, esto es equivalente a lo que se muestra en la gráfica de barras. En la tercera se muestra la cantidad de calorías consumidas por minuto por cada uno de los episodios del día y se ordenan cronológicamente y por colores indicando el tipo de episodio al que corresponden.

La utilidad de estas gráficas es combinar su información para conocer de un vistazo la actividad física del paciente. En este caso, como se aprecia en las figuras Figura 5.21 y Figura 5.22 el paciente realiza poco ejercicio, dado que la mayor parte del tiempo y del consumo se ocupa en episodios sedentarios, donde el consumo es casi tan bajo como en los episodios de sueño. Los episodios de actividad ligera son muy cortos y por tanto no llega a tener un gasto significativo de calorías como se refleja en la primera gráfica. Al contrario que un episodio de actividad moderada que aun siendo corto en duración su consumo es bastante elevado.

5.5 Manejo de los datos

Los datos de entrada deben estar en un fichero en formato csv y contener, como mínimo los siguientes campos:

- Sueño
- Clasificaciones_del_sueño
- Flujo_térmico_media
- Time
- Ligera
- Sedentaria
- Moderada
- Gasto_energético
- Acel_transversal_picos
- Acostado

El orden de las columnas no es relevante ya que los datos se obtienen por su nombre, por lo que es imprescindible mantener el nombre de los campos para un correcto funcionamiento de la aplicación.

Es posible, aunque poco frecuente, encontrar filas con datos inválidos, como tener algunos parámetros sin mediciones o fechas erróneas. Esto provoca fallos en los cálculos que se realizan en la aplicación por lo que son descartados mediante un filtro ejecutado tras leer todos los campos.

Por lo tanto, los datos resultantes de leer el fichero se componen de varias listas (una por campo) que contienen todas las filas válidas. Es posible que haya saltos de tiempo entre un dato y el siguiente debido a la falta de registros por la retirada del dispositivo o también tras aplicar el filtro anteriormente descrito. Esto no da ningún problema ya que las librerías gráficas empleadas son capaces de representar saltos temporales.

5.6 Descripción del código y arquitectura

Cada vez que se arranca el programa o se pulsa en el botón correspondiente para cargar los datos de un nuevo paciente, primero se llama a la clase *LectorFichero*.

LectorFichero recibe un fichero de datos en formato *csv* y se encarga de procesar todas las filas para descartar las inválidas y crear con ellos un objeto de la clase *Datos* que contiene una lista por cada parámetro del brazalete que va a ser de utilidad.

En esta aplicación se utilizan los siguientes parámetros:

- Tiempo: fecha completa de cada minuto registrado por el brazalete.
- Sueño: booleano que indica si el paciente está o no dormido.
- Clasificación de sueño: entero que indica el tipo de sueño en el que se encuentra el paciente, pudiendo tomar los valores 0, 2, 4 y 5 que corresponden a despierto, sueño ligero, sueño profundo y sueño muy profundo respectivamente.
- Flujo térmico (media)
- Temperatura cerca del cuerpo (media)
- Actividad sedentaria
- Actividad ligera
- Actividad moderada

- Gasto energético
- Aceleración transversal (picos)
- Acostado: booleano que indica si el paciente está o no tumbado.

La clase *Datos* también permite devolver una lista con los datos particionados en días completos, necesario para la pestaña de consumo energético.

Una vez se tienen guardados los datos pre-procesados, se pasan a la clase *SelEpisodio* que permite separar los datos en todos los tipos de episodios, a saber: sueños diurnos, sueños nocturnos, actividad sedentaria, actividad ligera y actividad moderada

La forma de dividir los datos en episodios consiste en recorrer cada fila del fichero determinando el tipo de episodio según la duración de sus parámetros y de la configuración de las interrupciones. Debido a que, por lo general a mayor intensidad de ejercicio, más corto es el episodio no se puede utilizar la misma configuración para todos los tipos de episodio porque de lo contrario perderíamos muchos de ellos o tendríamos demasiados muy pequeños.

La configuración por defecto es de:

- Episodios de sueño: máximo 35 minutos de interrupción
- Episodios de actividad sedentaria: 7 minutos de interrupción
- Episodios de actividad ligera: 4 minutos de interrupción
- Episodios de actividad moderada: 3 minutos de interrupción

Cuando un tipo de episodio supera su interrupción, se considera que el episodio ha finalizado y se siguen procesando las siguientes filas. Además, para que un episodio sea válido, debe durar un mínimo de 5 minutos.

La clase *SelEpisodio* recibe como parámetros los tipos de episodios que se desean obtener y también permite especificar si se quieren los episodios completos o no. Por defecto, los episodios vienen completos, es decir, cada episodio es un objeto del tipo *EpisodioCompleto* que hereda del tipo *Episodio* e incluye toda la información recolectada en *Datos* en el intervalo que dura el episodio. Si no hace falta tanta información, se puede desactivar el parámetro de episodios completos con lo que se obtienen objetos del tipo *Episodio* que únicamente incluyen el instante de inicio, el de fin, el tipo de episodio y su nombre (con la nomenclatura descrita).

Es posible filtrar los episodios por tipo en cualquier momento llamando a su método *update* que actualizará la lista de episodios filtrados con los parámetros especificados.

La aplicación se ejecuta desde el fichero *final.py* que crea una ventana personalizada de *pyqt* de la clase *VentanaMain* que utiliza las clases *LectorFichero* y *selEpisodio* para obtener los episodios y rellenar con ellos el contenido de la interfaz con las pestañas descritas en el apartado 5.1.

Para ello *VentanaMain* crea un objeto *Interprete*, un objeto *PanelScatter*, tres objetos *PanelSueno* y un objeto *PanelConsumo* que reciben sus respectivos elementos de la interfaz y los episodios necesarios.

La funcionalidad de cada pestaña se encuentra en su fichero correspondiente, de forma que el controlador de la pestaña Intérprete es *panelInterprete.py*, el de la pestaña Correlaciones es *panelScatter.py*, el de las tres pestañas de sueños es *panelSueno.py* y el de la pestaña Consumos es *panelConsumo.py*.

5.7 Tecnologías utilizadas

En este apartado vamos a describir brevemente algunas de las herramientas que hemos utilizado en el proyecto.

- **Paletton**[16]**:** Aplicación online que proporciona una herramienta para crear combinaciones y paletas de colores que funcionan bien juntos. La principal ventaja de Paletton es que no usa el espacio RGB común de los ordenadores, sino que es una rueda de color artística clásica. Aplica la teoría del color y funciona dentro de un espacio de color RYB especialmente creado para ella. Por tanto, las combinaciones de colores que se producen son diferentes de las que se pueden obtener en muchas aplicaciones gráficas conocidas.

- **Virtualenv**[17]**:** Es una herramienta para crear un espacio completamente independiente de otros entornos virtuales y de los paquetes instalados globalmente en el sistema. Un entorno virtual es una herramienta para mantener las dependencias requeridas por los diferentes proyectos en lugares separados. Soluciona el dilema de "el proyecto X depende de la versión 1.x, pero el proyecto Y necesita 4.x", y mantiene el directorio global de módulos limpio y manejable.

- **SciPy**[18]**:** es una librería de código abierto de herramientas y algoritmos matemáticos para Python.

- **Numpy**[19]**:** es una extensión de Python, que le agrega mayor soporte para vectores y matrices, constituyendo una librería de funciones matemáticas de alto nivel para operar con esos vectores o matrices.

- **Matplotlib**[20]**:** es una librería para la generación de gráficos a partir de datos contenidos en listas o arrays en el lenguaje de programación Python y su extensión matemática NumPy.

- **Mlpy**[21]**:** es un módulo de Python para el aprendizaje automático desarrollado sobre NumPy / SciPy y las Bibliotecas Científicas de GNU. Proporciona una amplia gama de métodos de aprendizaje automático con técnicas modernas.

[16] http://paletton.com/
[17] https://virtualenv.pypa.io/en/stable/
[18] http://www.scipy.org/
[19] http://www.numpy.org/
[20] http://matplotlib.org/
[21] http://mlpy.sourceforge.net/

Parte V

Conclusiones y trabajo futuro

6 Conclusiones

Al finalizar este proyecto se puede concluir que hemos cumplido el principal objetivo, es decir, desarrollar una herramienta que facilita la labor de los profesionales que trabajan con monitores de sueño. El proyecto empezó sin un enfoque claro y al final hemos sabido darle forma. El programa, dividido en varias fases, es capaz de leer, preprocesar, analizar y mostrar visualmente los datos utilizando las técnicas que hemos estudiado.

No podemos decir que seguimos un plan durante todo el proyecto ya que no lo teníamos debido a que al ser una colaboración entre dos mundos tan distintos como son la informática y la medicina llegar a un punto en común no siempre fue fácil. Aun así, creemos que el objetivo de mejorar el trabajo con el brazalete se ha completado. Tenemos que decir, de todas formas, que gracias a este proyecto tuvimos la oportunidad de vivir la experiencia de tener un cliente real y aprender todo lo que ello conlleva. Como tener cambios de requisitos continuos que pueden hacer que una idea siga adelante o se pierda en el olvido. Por otra parte, nos gusta saber que hemos realizado un proyecto que tendrá un fin real siendo utilizada para mejorar la vida de muchas personas.

Además, hemos completado una serie de metas que nos habíamos propuesto a nivel personal como son el aprendizaje de un nuevo lenguaje como es Python ya que teníamos un gran interés y ninguna experiencia previa en él; la realización de un proyecto desde cero poniendo mucho de nosotros en el resultado final y no solo siguiendo directrices; y por último queríamos emplear el conocimiento que habíamos adquirido en este tiempo.

7 Conclusions

Upon completion of this project we can conclude that we have fulfilled the main objective, i.e. to develop a tool that really facilitates the work of professionals working with sleep monitors. The project began without a clear approach and in the end we managed to shape it. The program, divided into several phases, is able to read, preprocess, analyze and visually display data using the techniques we have studied.

We can not say that we follow a plan throughout the project as we have not done it, because of the fact that it is a collaboration between two very different worlds such as computer science and medicine and reach a common point was not always easy. Nevertheless, we believe the goal of improving work with the sleep monitor is completed. We must say, however, that thanks to this project we had the opportunity to live the experience of having a real customer and we have learned all the aspects that customer interaction involves. Such as the constant changes of requirements that can make an idea evolve or dissapear. Moreover, we are pleased to have completed a project that will have a real purpose being used to improve the lives of many people.

In addition, we have completed a number of goals that we set personally, such as learning a new language as Python as we had a great interest and no previous experience in it; the realization of a project from scratch putting a lot of us in the final outcome and not just following guidelines; and finally we wanted to use the knowledge we have acquired during this time.

8 Trabajo futuro

A continuación, se citan algunas de las posibles mejoras que se nos han ocurrido.

- **Personalizar las opciones:** Ahora mismo la aplicación no permite demasiada configuración para el usuario medio. Configurar el tiempo mínimo de cada tipo de episodio o la duración máxima de una interrupción podría ser interesante de cara a encontrar anomalías en la vida del paciente por eso creemos que una mejora podría ser incluir una pestaña que permita esa configuración manual para que se pueda variar según las necesidades del usuario.

- **Soporte para clustering ampliado y detección de anomalías:** La idea es implementar la búsqueda de patrones o anomalías entre diferentes pacientes. Podría ser útil para descubrir irregularidades que no se ven analizando los datos de un único paciente.

- **Implementar detección de actividad intensa y muy intensa:** Creemos que sería una buena ampliación de la aplicación ya que se podría usar con otro tipo de personas con una actividad física mayor.

Parte VI

Organización del proyecto

9 Organización del proyecto

A continuación, explicaremos cómo nos hemos organizado, la metodología de trabajo que hemos seguido y lo que hemos hecho cada uno de nosotros en el proyecto.

9.1 Organización del proyecto

Empezamos a preparar el proyecto en verano cuando nuestro director nos mandó información sobre varios métodos de minería de datos que podríamos aplicar. Leímos documentación sobre una amplia variedad de conceptos y técnicas ya que todavía no teníamos claro el enfoque que se daría al proyecto. Aprendimos lo que es DTW (alineamiento temporal dinámico de dos series), SAX (simbolización de series), Hot SAX (búsqueda de anomalías en series temporales), clustering, distancias entre series y su posible utilidad para nosotros.

A finales de octubre, y tras las primeras reuniones con el equipo médico colaborador, empezamos a pensar en cómo realizar un trabajo de utilidad con las herramientas que se nos había proporcionado. Buscamos información sobre los posibles lenguajes de programación que se usan en minería de datos y tras leer varios artículos nos decidimos por Python dada su versatilidad.

No seguimos un plan estricto de tareas, sino que nos marcábamos metas tras cada reunión con el director. Estas metas casi siempre consistían en la primera etapa en estudiar la documentación e investigar por nuestra cuenta para poder preguntar dudas en la siguiente reunión, pero cuando empezamos a desarrollar las aplicaciones nos marcábamos un plazo hasta la siguiente reunión para que cada aplicación estuviera terminada. En este sentido el director nos daba bastante libertad para que nos organizáramos por nuestra cuenta.

En un principio no teníamos claro cuál sería la finalidad del proyecto ni de la posible aplicación. Contábamos con el brazalete, el software del fabricante y unos pocos ficheros de datos registrados por el dispositivo. El doctor y la enfermera aseguraban que podríamos sacar mucha información del dispositivo que no se mostraba en el software. Así que empezamos por investigar con estos datos para conocer la funcionalidad del programa, ver qué podríamos aportar y empezar a entender los registros almacenados.

Empezamos creando una aplicación para nuestro uso donde podíamos representar cualquier parámetro registrado por el brazalete en una sola gráfica, ya que el software sólo lo podíamos usar en las oficinas de Ibermutuamur. Solapamos varios parámetros en la misma gráfica para observar e intentar encontrar algún fenómeno, patrón o anomalía que se pudiera dar en instantes determinados. Dadas las pautas recibidas en Ibermutuamur nos centramos en aquellos instantes en los que el paciente dormía, observando el comportamiento de los demás parámetros al despertarse como las fases de sueño, temperatura, consumo y flujo térmico. Y comprobamos que efectivamente la temperatura y el flujo térmico están relacionados ya que tienen formas inversas en los mismos instantes de tiempo y esto se repite en la mayoría del tiempo de sueño.

A continuación, decidimos realizar una segunda aplicación para filtrar de los datos por episodios de sueño y actividad física y mostrarlos en una única gráfica. Esto en principio nos permitía contemplar de un vistazo cada episodio sin tener que navegar por toda la gráfica buscando dónde se encuentra cada uno y nos pareció adecuado para la aplicación final por su facilidad de uso y rapidez. Observamos que el mismo fenómeno se producía en episodios de actividad física solo que a distintas temperaturas, por lo que después incluimos un diagrama de dispersión representando la temperatura frente al flujo térmico del episodio seleccionado. De esta manera se puede apreciar mejor la relación

entre ambos parámetros. Finalmente calculamos el coeficiente de correlación de cada episodio y lo visualizamos junto con otros datos de la gráfica para confirmar finalmente que al aumentar el flujo térmico disminuye la temperatura y viceversa.

Tras realizar ambos programas los presentamos en Ibermutuamur para comprobar si íbamos bien encaminados y para escuchar cualquier comentario que nos quisieran hacer sobre los programas. Nos guiaron para centrar el resto de la aplicación por dos vías, el análisis de los episodios de sueño y el análisis del consumo energético consumido según la actividad física.

Conocida la relación entre la temperatura y el flujo, probamos a clasificar los episodios de sueño mediante clustering jerárquico para ver si podíamos encontrar alguna pista que nos indicase un patrón, repetición o alguna anomalía entre grupos de sueños similares. Visto que el agrupamiento por clustering parecía funcionar correctamente y tras varias reuniones entre las cuales íbamos perfeccionando las aplicaciones anteriores, empezamos una nueva aplicación donde queríamos mostrar de la forma más sencilla posible los resultados del agrupamiento entre episodios por temperatura y flujo y entre consumos energéticos. Al mismo tiempo, con las recomendaciones del director empezamos otro programa para el análisis del consumo, que trataría de resumir de forma visual los consumos energéticos totales del paciente, organizados por días y tipo de actividad física y también por horas de inicio y fin de cada actividad. De esta manera y de un vistazo se podría apreciar cómo de sedentario o activo es el estilo de vida del paciente sin tener que consultar los datos al minuto de cada día.

En otra reunión en Ibermutuamur presentamos una versión inicial de estas dos últimas aplicaciones para evaluar los problemas que fueran surgiendo con su manipulación y las posibles carencias que pudiera haber. A partir de ese momento y con las sugerencias obtenidas, decidimos empezar a realizar el programa final incluyendo en él todos los programas anteriormente descritos con nuevas funciones y los arreglos pertinentes.

9.2 Contribución al proyecto

Este apartado sirve para explicar lo que ha hecho cada uno en el proyecto.

9.2.1 Álvaro Allegue Lorenzo

Las siguientes tareas han sido realizadas en mayor o menor medida entre mi compañera y yo. Dada la dificultad de diferenciar los trabajos por separado, expongo una lista con las tareas en las que he participado:

- Lectura de la documentación proporcionada por el director sobre técnicas de minería de datos.

- Búsqueda de información sobre el dispositivo proporcionado. Al tratarse de un equipo nuevo tanto para el departamento de ibermutuamur como para nosotros y dado que el fabricante no respondía, dejamos algunas dudas sin resolver. Sin embargo, no hemos tenido ningún impedimento para obtener la información necesaria, ya sea por parte del departamento como documentándonos por Internet.

- Análisis de funcionalidad del software del fabricante. No disponíamos del programa fuera de las instalaciones de Ibermutuamur, por lo que era necesario acudir allí para recopilar toda la información que pudiese otorgar.

- Documentación de temas médicos relacionados con trastornos de sueño, metabolismo y

ejercicio físico. Para ello contamos con la ayuda y el conocimiento del doctor y la enfermera además de otras fuentes.

- Elección del lenguaje de programación a utilizar, Python 2.7 y aprendizaje a través de la web CodeCademy y la documentación.

- Elección y aprendizaje de un sistema de control de versiones (Github) con el que trabajar en un repositorio común.

- Búsqueda de librerías para implementar interfaces de usuario y de un programa que facilite la tarea de colocar los elementos en la interfaz. Como herramienta de diseño elegimos QtDesigner y como librería PyQt por ser ampliamente utilizada, disponer de buena documentación y ejemplos y ser compatible con buena cantidad de librerías que íbamos a utilizar.

- Búsqueda de librerías para implementar gráficas con posibilidad de interacción por parte del usuario y que sean suficientemente personalizables. Elegimos PyQtGraph y matplotlib. La primera por su gran interacción y rendimiento y la segunda por ser altamente personalizable y disponer de una buena documentación.

- Búsqueda de librerías para operaciones matemáticas en general, cálculo de distancias en series temporales y clustering. Elegimos Numpy, Mlpy y Scipy.

- Elección de las herramientas de desarrollo. En principio elegimos Ubuntu como sistema operativo ya que la mayoría de la documentación y ejemplos usaban Linux y aunque la plataforma objetivo era Windows, había programas para exportar un ejecutable a dicho sistema operativo. Como entorno de desarrollo la mayoría del proyecto lo hicimos con Geany aunque la aplicación final la desarrollamos en Eclipse con el plugin PyDev que nos permitía depurar con más facilidad.

- Realización de pruebas con las nuevas librerías que íbamos añadiendo. Las pruebas, además de ser útiles para aprender a manejarlas y dada la gran cantidad a utilizar, eran imprescindibles para averiguar la compatibilidad entre ellas. Esto nos hizo descartar algunas por su incompatibilidad con el resto.

- Implementación de una primera aplicación que muestre todos los datos de un paciente. En un principio era para uso propio con el fin de visualizar los datos sin depender del software del fabricante.

- Implementación de una segunda aplicación que muestre los datos particionados en tipos de episodios. Inicialmente su utilidad era la de comparar episodios para averiguar qué relación tenían los parámetros registrados por el dispositivo. Tras observar la relación entre la temperatura y el flujo térmico, añadimos gráficas de dispersión y el coeficiente de correlación para dar constancia de esta relación. Posteriormente se mejoró la visualización al añadir coloreado de los episodios y una nueva distribución de la información.

- Implementación de la tercera aplicación donde se pretendía agrupar los episodios de sueño mediante las técnicas de clustering estudiadas y poder así identificar los episodios que son diferentes al resto. También fue añadida a la aplicación final.

- Implementación de la cuarta aplicación con la idea de mejorar la presentación de los datos relacionados con la actividad física y el consumo calórico.

- Diseño e implementación de la aplicación final adaptando e incorporando las aplicaciones que fuimos desarrollando a lo largo del proyecto.

- Revisión del funcionamiento de la aplicación e implementación de mejoras y correcciones debidas.

- Buscar la forma de exportar el programa como un ejecutable de Windows, que pueda ejecutar la aplicación sin necesidad de instalar programas externos. Tras probar con varias aplicaciones en Linux sin conseguir resultados funcionales, decidimos pasar nuestro programa a Windows y emplear Virtualenv. Este paso dió bastantes problemas pues no todas las librerías funcionaban en Windows y hubo que revisar toda la aplicación. Finalmente tenemos la aplicación instalable en Windows mediante un entorno virtual que contiene todas las librerías necesarias y es de fácil instalación y desinstalación.

- Dada mi mayor disponibilidad por compatibilidad de horario solía acudir a las reuniones en Ibermutuamur con el doctor y la enfermera cuando mi compañera no podía para mostrar los avances, consultar dudas y acordar cambios y nuevas funcionalidades del programa.

Respecto a la realización de la memoria dividimos el trabajo y yo me encargué principalmente de:

- Redacción del apartado Introducción.

- Documentación sobre los dispositivos en la actualidad y redacción de su introducción.

- Redacción del apartado Dispositivo utilizado.

- Redacción de la mayor parte de funcionalidad de la aplicación, manejo de los datos y descripción del código.

- Redacción de parte de la organización del proyecto.

- Revisión del contenido de la memoria

9.2.2 Ana Carolina Rueda Silva

Una vez decidido el lenguaje que íbamos a usar en el proyecto, mi primera tarea fue la de aprender Python. Para ello, seguí un par de cursos en Internet, uno en CodeCademy[22] y otro en Coursera[23]. Las primeras lecciones fueron sobre sintaxis y la creación de una calculadora básica. Pero pronto empecé a probar con fechas, entrada y salida y strings. Sin olvidarme de las listas, diccionarios, creación de funciones y clases que me vendrían muy bien para la aplicación. Para finalizar, siguiendo el tutorial de CodeCademy cree una versión simplificada del juego *Hundir la flota* con el que practiqué todo lo que había ido aprendiendo.

Tras esto dediqué tiempo a buscar librerías gráficas para ahorrarnos el trabajo de crear nosotros manualmente las gráficas. Llegamos a la conclusión de que Plotly[24] sería la mejor opción, pero pocos días después tuvimos que volver a buscar ya que Plotly es una librería online y no

[22] https://www.codecademy.com/
[23] https://www.coursera.org/learn/interactive-python-1
[24] https://plot.ly/

estábamos seguros de que fuera la mejor opción dado que era posible que trabajáramos con datos privados. Así, llegamos a PyQtGraph[25] que finalmente fue la librería que usamos en la primera aplicación.

Una vez que teníamos elegida la librería grafica empecé a aprender Git en CodeCademy para poder usar el sistema de control de versiones Git con un repositorio público en GitHub y mantener todo el trabajo ordenado.

Para empezar a trabajar en la primera aplicación tuvimos que buscar una herramienta para diseñar interfaces de usuario de forma rápida. Instalar Python y varias librerías que nos harían falta. A partir de este punto nos dividimos el trabajo. Yo me encargaba de leer los ficheros csv para cargar los datos que íbamos a analizar y para ello utilicé la librería NumPy que permite trabajar con vectores y matrices de forma cómoda. La primera versión fue bastante ingenua ya que cargaba solo las columnas de datos que nos hacían falta marcando exactamente el número de cada columna según el fichero de prueba que usábamos. Así pues, solo leía las columnas de tiempo, tipo de sueño, tres de tipo de actividad (sedentaria, ligera y moderada), consumo energético, actividad física, temperatura, flujo térmico y acelerómetro transversal.

Tras esto, hice un filtro sencillo para separar los minutos de sueño de los de vigilia porque buscábamos centrarnos en el sueño. Para finalizar mi trabajo en la aplicación elegí, junto a mi compañero, los colores con los que mostrar las gráficas y creé un método que coloreaba los datos. Elegimos tonos azules para el sueño y verde para las actividades, además de usar verde, naranja y blanco para las otras gráficas. Esta aplicación fue un buen inicio dado que nos permitía *jugar* con los datos y verlos gráficamente para así decidir qué podría ser útil representar y qué no aportaba demasiada información.

Después de un par de reuniones con el director y el doctor decidimos quitar la gráfica de la actividad física ya que no aportaba nada y empezamos a diseñar la segunda aplicación para buscar relaciones entre el flujo térmico y la temperatura durante cualquier tipo de actividad.

Con esta aplicación tuve que cambiar la forma de leer los ficheros ya que vimos que según se configure el brazalete las columnas de datos podían variar su posición con lo que lo que teníamos hecho fallaba. Decidí que lo mejor sería leer cada columna por su nombre para que la configuración del brazalete no supusiera ningún problema si se cambiaba de un paciente a otro. Esto me generó algún problema por la codificación de caracteres con la que la aplicación del fabricante exporta los ficheros csv. Principalmente por los caracteres especiales que no existen en inglés como la ñ y las tildes debido a que Python no puede trabajar con ellos.

Para esta aplicación nos hacía falta una partición más fina de los distintos tipos de actividad y episodios de sueño por lo que tuve que crear un nuevo método para hacerlo. Esta parte fue algo complicada ya que tenía que tener en cuenta cuatro tipos de datos distintos al mismo tiempo y las posibles interrupciones entre episodios. La primera versión que hice no funcionaba ya que creaba los índices de cada episodio de forma separada, los sueños por un lado y las actividades por otro. Tenía un método que se encargaba de unirlo todo, pero fallaba por la irregularidad de los datos así que tuve que pensar en otro que fuera un "todo en uno". Al final lo resolví con un método algo complicado que funcionaba para todos los casos con los que pudimos probar a pesar de que no tenía un control de interrupciones demasiado fino ya que usaba el mismo tiempo para todos los tipos de episodios.

Fue en este momento cuando decidimos, junto al director, que lo mejor sería que todas las aplicaciones fueran consistentes en cuanto al color de las gráficas para facilitar las cosas al usuario.

[25] http://www.pyqtgraph.org/

Cambié los colores de la primera aplicación para que fueran más ilustrativos usando Paletton y creamos una clase donde guardarlos todos para unificarlos en todas las aplicaciones que desarrolláramos. Los colores esta vez los puse en diferentes tonos de azul para el sueño y desde amarillo a rojo para las distintas actividades. Así conseguí que se distinguiera mejor cada actividad y que se destacara más el cambio de una a otra. También necesitábamos una nueva librería que nos ayudara a hacer los cálculos de correlación y los diagramas de dispersión. Decidimos, después de una búsqueda rápida, que lo mejor sería utilizar matplotlib por todo lo que ofrecía para gráficos en 2D y SciPy para los cálculos.

Cuando terminamos la aplicación tuvimos nuevamente una reunión con el doctor para presentar el trabajo realizado y para que nos indicara lo que le resultaba útil y lo que no. También creamos un mini manual sobre las dos aplicaciones para que se familiarizara con ellas antes de la reunión y pudiera preguntarnos dudas o sugerirnos cambios. Con esta información tuvimos varias reuniones más con el director donde vimos los puntos a corregir de las dos aplicaciones y diseñamos una tercera aplicación con la que clasificar los episodios de sueño para poder compararlos ya que a todos nos parecía interesante. También vimos que podría ser útil crear una cuarta aplicación que presentara un resumen de los datos mejorando el informe estático con el que trabajaba el doctor.

En esta tercera aplicación tendríamos que utilizar algunas de las técnicas sobre las que habíamos leído tiempo atrás por lo que volví a leer sobre ellas para recordarlas y empezamos a buscar implementaciones de DTW y clustering en Python que pudiéramos usar. Descubrimos que podíamos aprovechar la librería SciPy ya que incorporaba métodos para hacer clustering y además usar mlpy para el cálculo de las distancias con DTW.

De nuevo tendría que cambiar la forma en que se dividían los episodios para que se hiciera de una forma más exhaustiva ya que necesitábamos dar distintos énfasis a las actividades. Nos habían dicho que una actividad moderada o intensa era muy importante, aunque durase pocos minutos. Antes de ponerme a ello dediqué un par de días a pensar en cuál sería la mejor forma de hacerlo para no tener que cambiarlo de nuevo. Decidí que lo mejor sería empezar de cero y crear episodios de cada actividad según me los encontrara, aunque durasen un minuto, para a continuación fusionarlos teniendo en cuenta las interferencias que ahora serían distintas para cada tipo episodio dada la distinta importancia que se daba a unos y a otros. Por esto también creé varios métodos para incluir los episodios más cortos (interferencias) dentro de otros mucho más largos y agruparlos. Hice asimismo un método para poner nombre a cada episodio según el tipo de actividad para que al mostrarlos en las aplicaciones fuera más fácil reconocerlos. Tuve que hacer muchas pruebas para asegurarme de que funcionaba bien la creación de episodios.

La cuarta aplicación me obligaba a modificar la lectura de ficheros ya que no tenía en cuenta los días. Así pues, tenía que crear una división por días además de filtrar los datos anómalos (para la anterior). Para ello tuve que buscar la manera de trabajar con el formato de fechas que guarda el brazalete. Descubrí que guarda la hora en formato UNIX con lo que solo necesitaba transformar esos segundos a un formato fecha. Para ello utilicé el módulo datetime[26] de Python que crea un objeto datetime que me facilitaba la labor de crear la lista de días. Con las fechas terminadas cambié la forma en que se ponían los nombres de los episodios para añadir el día y crear un formato único a todas las aplicaciones.

Tuvimos una reunión con el director para enseñarle las nuevas aplicaciones y que nos diera su opinión. Nos propuso varios cambios ya que representábamos información inútil en la matriz de distancias y nos dio la idea de mostrar franjas verticales para destacar el tiempo de sueño y asimismo nos sugirió cambios en la organización de la tercera aplicación. También decidimos cambiar la forma

[26] https://docs.python.org/2/library/datetime.html

de representar la serie temporal en la cuarta aplicación ya que la primera versión era poco informativa. Finalmente hicimos un manual donde se explicaban estas aplicaciones y cómo interpretar los gráficos y el dendrograma para presentársela al doctor en la siguiente reunión.

Teniendo en cuenta los cambios que nos sugirieron en estas reuniones empezamos a crear la última aplicación que englobaría a todas las demás, pero separando la de clustering en tres pestañas distintas para analizar los sueños diurnos y los nocturnos por separado. Mejoramos la información que se muestra en cada pestaña y cambiamos el diseño de las pestañas para que toda la aplicación sea consistente. También limpiamos y mejoramos todo el código que llevábamos escrito y lo ordenamos mejor aprovechando que hacer la aplicación final era casi como empezar de nuevo. Cuando tuvimos una versión más o menos completa empezamos a investigar la forma de instalarla en Windows ya que se usaría en ese sistema operativo. Esto nos hizo retroceder varios pasos porque la habíamos desarrollado en Ubuntu y nunca habíamos hecho pruebas en Windows con lo que cuando conseguimos ejecutarla nos daba varios fallos, como no mostrar los puntos de tiempo en las gráficas por la forma de tratar las fechas. Asimismo, nos dio la oportunidad de hacer varias pruebas a la aplicación buscando fallos que pudiéramos haber pasado por alto y descubrimos que teníamos un problema con ciertos ordenadores al usar un tamaño fijo de ventana. Por eso, decidimos implementar frames para que la aplicación se adaptara a cualquier resolución. No conseguimos crear un ejecutable que funcionara en Windows debido a las dependencias con gran cantidad de librerías y programas externos por lo que al final optamos por crear un entorno virtual utilizando virtualenv, lo que nos vino bien dado que permite usar la aplicación, aunque se use un ordenador con funciones limitadas para el usuario. Cuando lo probamos en varios ordenadores de familiares y amigos y todo funcionaba bien me dispuse a crear un manual de instalación intentando que fuera lo más sencillo posible para los usuarios finales.

En cuanto a la memoria, me encargaba de dar formato y estructura. Escribir el resumen y parte de la introducción, buscar información sobre los distintos dispositivos monitores de sueño para completar el trabajo de esta sección iniciado por Álvaro y organizar todo ese apartado. Igualmente dediqué tiempo a investigar sobre los métodos de minería de datos y en especial de minería de datos de series temporales y escribí sobre ello. Me ocupé de explicar las pestañas de clustering de la aplicación y escribir sobre las tecnologías utilizadas y las conclusiones y el trabajo futuro. Asimismo, tuve que redactar mi contribución y parte de la organización del proyecto. También hice la bibliografía, los anexos y los índices. Sin olvidarme de revisar toda la memoria un par de veces.

En general ambos hemos colaborado en mayor o menor medida en todos los grandes bloques del proyecto y, entre los dos, hemos solucionado los problemas y conflictos que iban surgiendo hasta conseguir hacer real una primera idea que parecía muy lejana. El trabajo de ambos sirvió no solo para conseguir terminar un trabajo de fin de grado, sino que también nos sirvió para aprender a desarrollar un proyecto desde cero y a lidiar con los conflictos que se iban planteando a lo largo del mismo.

Parte VII

Anexos

A. Ética y privacidad de los datos

La primera cuestión que se plantea es la vulnerabilidad de la privacidad, a la posibilidad de obtener información que afecte de modo significativo la esfera de personalidad e incluso que sea capaz de proporcionar herramientas de control social. Por eso es fundamental recordar el derecho a la protección de datos personales de la Sentencia 292/2000. Esta facultad de control se proyecta sobre todos nuestros datos, ya que lo relevante no es su carácter público o privado sino la información que podamos extraer a partir de su tratamiento.

Tenemos que decidir si en realidad existen o no datos personales. Un dato es «cualquier información numérica, alfabética, gráfica, fotográfica, acústica o de cualquier otro tipo concerniente a personas físicas identificadas o identificables». Así la forma de eludir la aplicación de la normativa es bien simple: anonimizar, disociar los datos de manera tal que no se nos permita la identificación de un afectado o interesado. Sin embargo, la cuestión no es tan sencilla y en la práctica debemos distinguir diferentes aspectos.

Anonimización de los datos

En esta fase se deben desligar completamente los datos de sus titulares de modo que resulten imposibles de vincular, aunque esta no es una operación sencilla ni banal. La anonimización constituye un tratamiento en sí misma y como tal debería ser compatible con el tratamiento original y en base a ella contar con un fundamento legal que la legitime.

En esencia la anonimización exige:

- Que no pueda ser establecido vínculo alguno entre el dato y su titular sin un esfuerzo desproporcionado
- Que sea irreversible
- Que en la práctica sea equivalente al de un borrado permanente

La calidad de los datos

En la recogida directa de datos personales el responsable tiene la legítima confianza en que el afectado le facilitará datos veraces. Cuando se habla de patrones y predictibilidad se debe confiar en que el patrón sea fiable y lleve a conclusiones adecuadas, ya que de lo contrario el principio de veracidad de los datos peligrará.

La finalidad

El propósito de los datos puede quedar anticuado, pero no se agota con el uso. Por tanto, es susceptible de ser reutilizado alterando el concepto de finalidad, ya no durante su uso o respecto de nuestra decisión respecto de este, sino ante resultados inesperados.

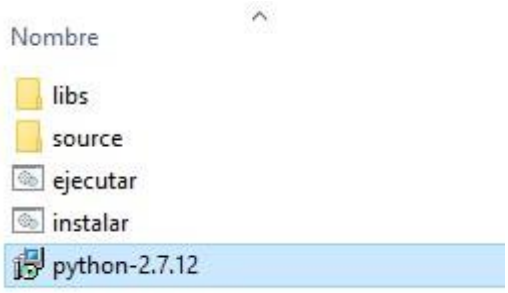
Los datos empleados en este proyecto son de particulares voluntarios, por tanto, por nuestra parte no serán conservados en forma que permita la identificación del interesado durante un período superior al necesario para los fines en base a los cuales han sido recabados o registrados.

B. Manual de instalación

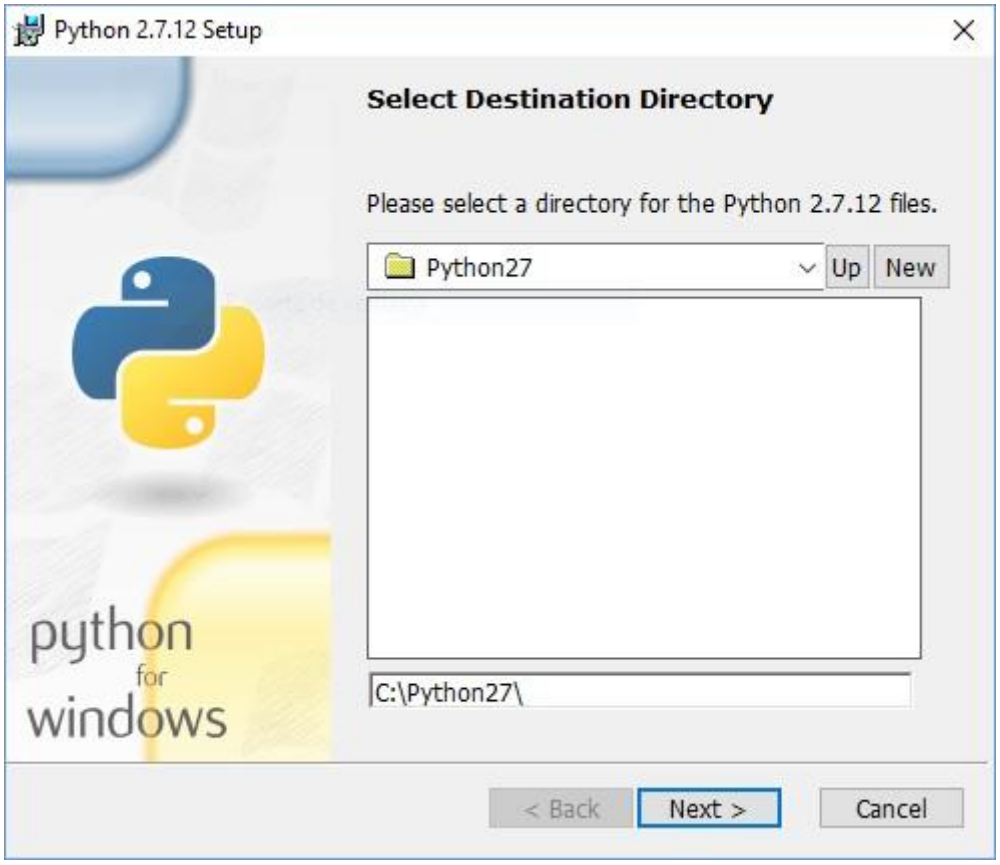
Instalación de Python

- Este paso se puede omitir si ya se dispone de Python 2.7 instalado.

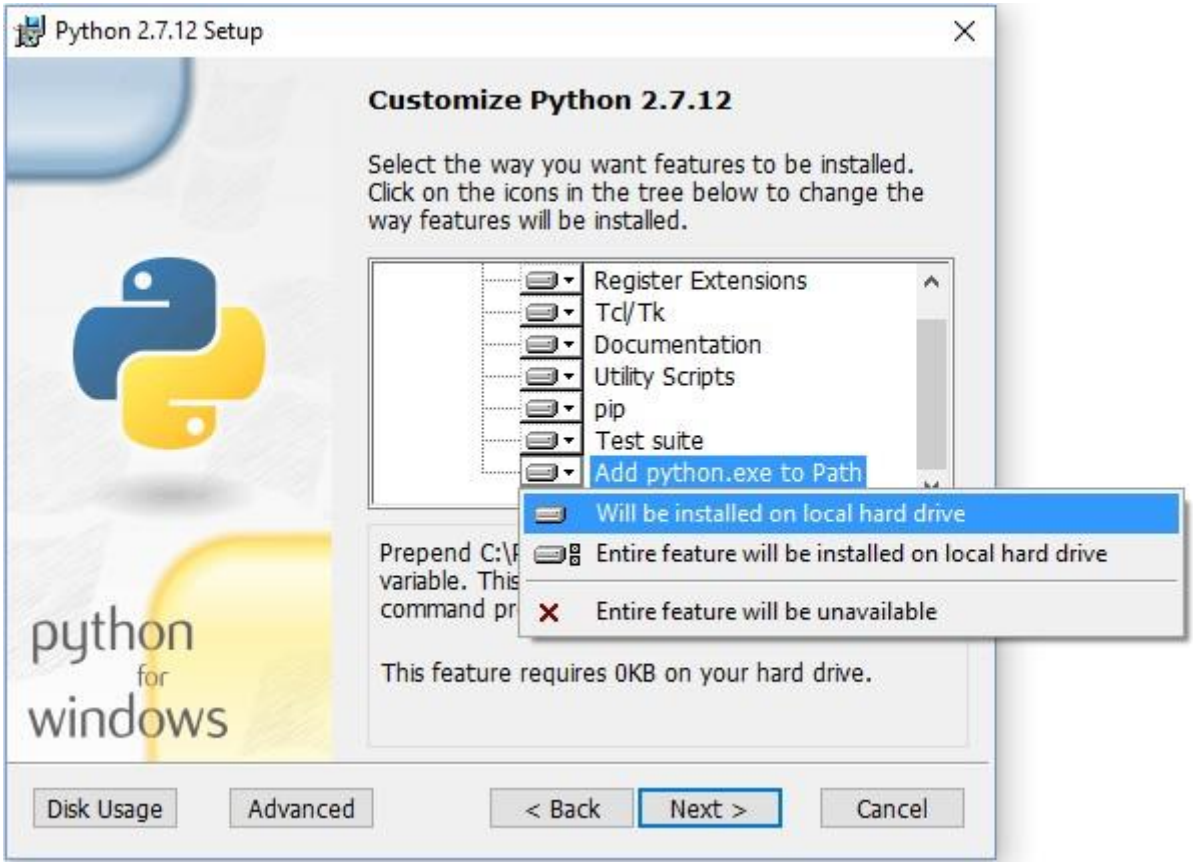
Buscar el fichero de instalación de Python en la carpeta que se proporciona y hacer doble clic para empezar su ejecución.



Saldrá una ventana como la que se ve en la Ilustración 1, una vez elegida la ruta de instalación pulsar el botón “next” hasta llegar a la ventana de la Ilustración 2.



A continuación, activar la última opción de personalización como se muestra en la Ilustración 2.

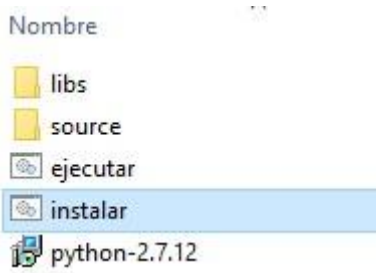


Pulsar “next” y aceptar la autorización de ejecución para que se inicie la instalación. Finaliza en cuanto se muestra la Ilustración 3 y se puede pulsar el botón “Finish”.

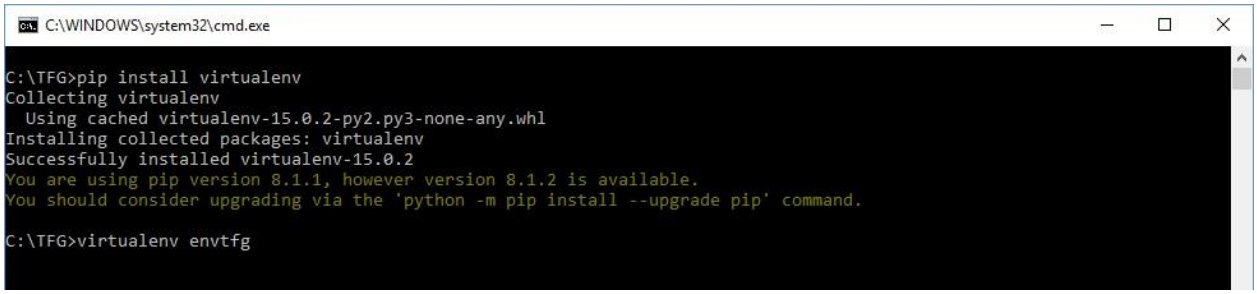
- Es importante **reiniciar** el equipo tras la instalación de Python.

Instalación del entorno virtual

Buscar el fichero de instalación del entorno virtual en la carpeta que se proporciona y hacer doble clic.



Se mostrará la siguiente ventana.

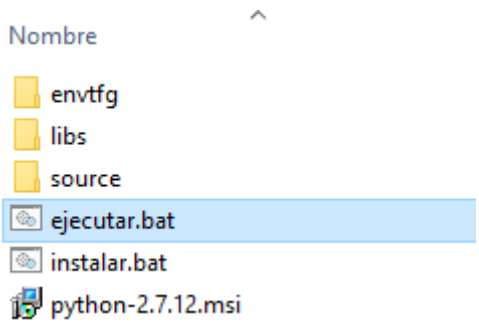


- Este paso puede **tardar** varios minutos.

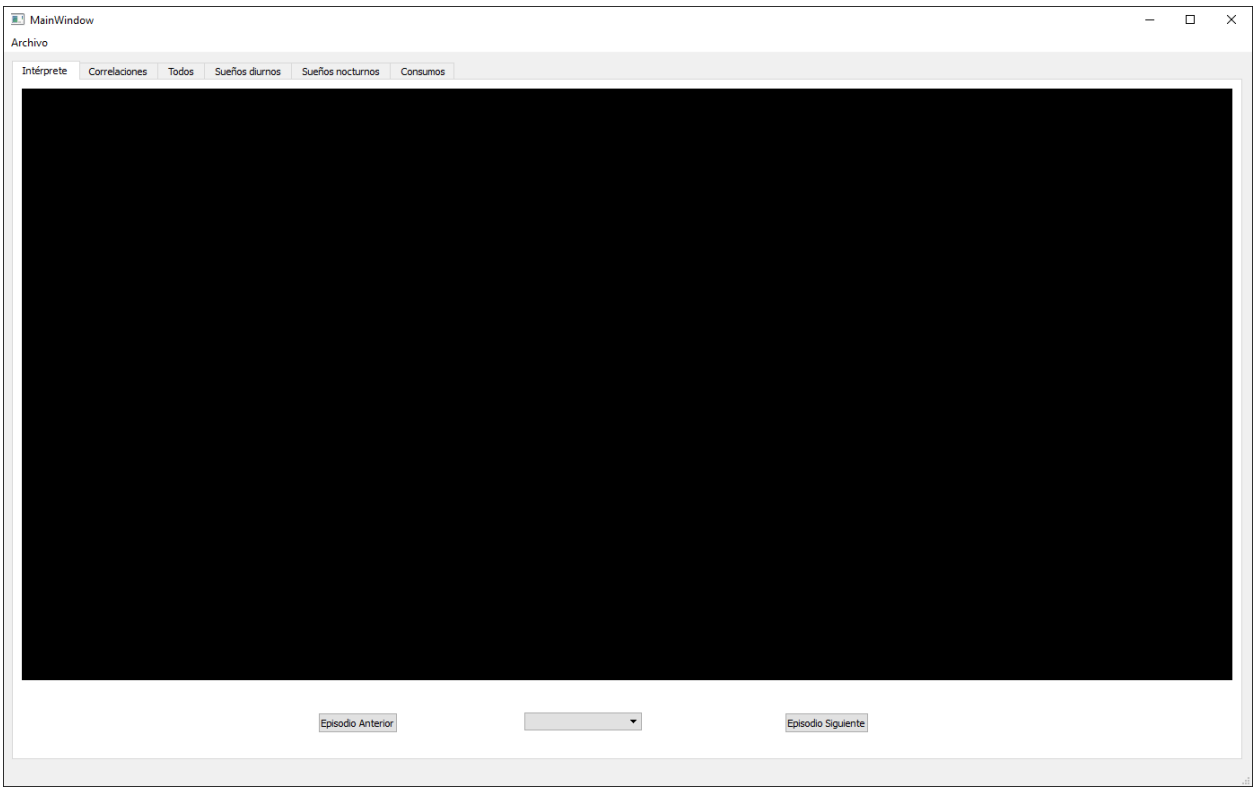
Al finalizar, la ventana se cierra sola y se habrá creado la carpeta “envtfg”:

Ejecución

Buscar el fichero “ejecutar” en la carpeta proporcionada y hacer doble clic.



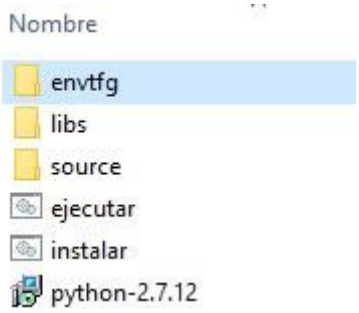
Tras ejecutar la aplicación se mostrará la ventana siguiente donde se pueden cargar datos desde la pestaña Archivo -> Abrir para empezar a trabajar.



Esta aplicación solo admite datos en formato *.csv

Desinstalación

Para desinstalar la aplicación basta con borrar la carpeta “envtfg” creada en la instalación.



Bibliografía

[1] P. D. Loprinzi y B. J. Cardinal, «Association between objectively-measured physical activity and sleep, NHANES 2005–2006», *Mental Health and Physical Activity*, vol. 4, n.º 2, pp. 65-69, dic. 2011.

[2] K. G. Baron, K. J. Reid, y P. C. Zee, «Exercise to improve sleep in insomnia: exploration of the bidirectional effects», *Journal of clinical sleep medicine: JCSM: official publication of the American Academy of Sleep Medicine*, vol. 9, n.º 8, pp. 819-824, ago. 2013.

[3] M. A. Kredlow, M. C. Capozzoli, B. A. Hearon, A. W. Calkins, y M. W. Otto, «The effects of physical activity on sleep: a meta-analytic review», *J Behav Med*, vol. 38, n.º 3, pp. 427-449, ene. 2015.

[4] J. M. Kelly, R. E. Strecker, y M. T. Bianchi, «Recent Developments in Home Sleep-Monitoring Devices», *ISRN Neurol*, vol. 2012, oct. 2012.

[5] «Zeo, el gadget más esperado por los insomnes.», *El Confidencial*. [En línea]. Disponible en: http://www.elconfidencial.com/tecnologia/2013-03-02/zeo-el-gadget-mas-esperado-por-los-insomnes_767891/. [Accedido: 01-ago-2016].

[6] «SleepImage, wearable para tratar la apnea del sueño», *Paperblog*. [En línea]. Disponible en: http://es.paperblog.com/sleepimage-wearable-para-tratar-la-apnea-del-sueno-3061999/. [Accedido: 01-ago-2016].

[7] «SleepHealth debuts as first ResearchKit app & study to support IBM Watson Health Cloud», *AppleInsider*, 01-mar-2016. [En línea]. Disponible en: http://appleinsider.com/articles/16/03/02/sleephealth-debuts-as-first-researchkit-app-study-to-support-ibm-watson-health-cloud. [Accedido: 07-ago-2016].

[8] «Monitorización de pacientes mediante camisetas inteligentes», *Web Oficial de la Universidad Carlos III de Madrid*, 08-oct-2007. [En línea]. Disponible en: http://portal.uc3m.es/portal/page/portal/actualidad_cientifica/noticias/camiseta_inteligente. [Accedido: 05-ago-2016].

[9] G. López, V. Custodio, y J. I. Moreno, «LOBIN: E-Textile and Wireless-Sensor-Network-Based Platform for Healthcare Monitoring in Future Hospital Environments», *IEEE Transactions on Information Technology in Biomedicine*, vol. 14, n.º 6, pp. 1446-1458, nov. 2010.

[10] «Sleep-tracking apps: Can they really help you rest easy?», *Washington Post*. [En línea]. Disponible en: https://www.washingtonpost.com/news/the-switch/wp/2014/09/22/sleep-tracking-apps-can-they-really-help-you-rest-easy/. [Accedido: 05-ago-2016].

[11] «The SenseWear Armband», *Wearable Metabolic, Sleep and Activity Monitoring*, 10-jun-2011. [En línea]. Disponible en: https://templehealthcare.wordpress.com/the-sensewear-armband/. [Accedido: 27-jul-2016].

[12] «Monitor metabolico Bodymedia SenseWear Armband». [En línea]. Disponible en: http://www.adieta.com/holter-metabolico/bodymedia/monitor-metabolico-bodymedia-sensewear-armband~1174.html. [Accedido: 27-jul-2016].

[13] «Temple Healthcare: BodyMedia». [En línea]. Disponible en: http://www.templehealthcare.com.au/pgs-vis/bodymedia/product-bodymedia.html. [Accedido: 27-jul-2016].

[14] «Data mining», *Wikipedia, the free encyclopedia*, 16-jul-2016. [En línea]. Disponible en: https://en.wikipedia.org/w/index.php?title=Data_mining&oldid=730044451. [Accedido: 26-jul-2016].

[15] C. A. Ratanamahatana, J. Lin, D. Gunopulos, E. Keogh, M. Vlachos, y G. Das, «Mining Time Series Data», en *Data Mining and Knowledge Discovery Handbook*, O. Maimon y L. Rokach, Eds. Springer US, 2009, pp. 1049-1077.

[16] K. Chakrabarti, E. Keogh, S. Mehrotra, y M. Pazzani, «Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases», *ACM Transactions on Database Systems*, vol. 27, n.º 2, pp. 188–228, jun. 2002.

[17] E. J. Keogh y M. J. Pazzani, «An Enhanced Representation of Time Series Which Allows Fast and Accurate Classification, Clustering and Relevance Feedback.», presentado en Proceedings of the 4th Int'l Conference on Knowledge Discovery and Data Mining, 1998, vol. 98, pp. 239–243.

[18] J. R. Quinlan, *C4.5: programs for machine learning*. San Mateo, Calif.: Morgan Kaufmann Publishers, 1993.

[19] «Nearest neighbour classifiers», *Wikipedia, the free encyclopedia*, 18-feb-2016. [En línea]. Disponible en: https://en.wikipedia.org/w/index.php?title=Nearest_neighbour_classifiers&oldid=705568161. [Accedido: 17-ago-2016].

[20] P. Indyk, N. Koudas, y S. Muthukrishnan, «Identifying Representative Trends in Massive Time Series Data Sets Using Sketches», en *Proceedings of the 26th International Conference on Very Large Data Bases*, San Francisco, CA, USA, 2000, pp. 363–372.

[21] E. Keogh, S. Lonardi, y B. «Yuan-chi» Chiu, «Finding Surprising Patterns in a Time Series Database in Linear Time and Space», en *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, 2002, pp. 550–556.

[22] S. Zhang, C. Zhang, y Q. Yang, «Data preparation for data mining», *Applied Artificial Intelligence*, vol. 17, n.º 5-6, pp. 375-381, may 2003.

[23] F. Herrera, J. Riquelme, y R. Ruiz, «Preprocesamiento de Datos», presentado en Reunión Red Nacional DM & ML, 2004.

[24] «Feature scaling», *Wikipedia, the free encyclopedia*, 22-ago-2016. [En línea]. Disponible en: https://en.wikipedia.org/w/index.php?title=Feature_scaling&oldid=735723423. [Accedido: 20-ago-2016].

[25] P. Juszczak, D. M. J. Tax, y R. P. W. Duin, «Feature Scaling in Support Vector Data Description», *ResearchGate*, may 2002.

[26] R. Agrawal, C. Faloutsos, y A. N. Swami, «Efficient Similarity Search In Sequence Databases», en *Proceedings of the 4th International Conference on Foundations of Data Organization and Algorithms*, London, UK, UK, 1993, pp. 69–84.

[27] D. Q. Goldin y P. C. Kanellakis, «On Similarity Queries for Time-Series Data: Constraint Specification and Implementation», presentado en Proceedings of the 1st International Conference on Principles and Practice of Constraint Programming (CP'95), 1995, pp. 137-153.

[28] R. Agrawal, K.-I. Lin, H. S. Sawhney, y K. Shim, «Fast similarity search in the presence of noise, scaling, and translation in times-series databases», presentado en Proceedings of the 21st International Conference on Very Large Data Bases (VLDB'95), 1995, pp. 490-501.

[29] E. Keogh y S. Kasetty, «On the need for time series data mining benchmarks: A survey and empirical demonstration.», presentado en Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'02), 2002, pp. 102-111.

[30] E. Keogh, T. Palpanas, V. B. Zordan, D. Gunopulos, y M. Cardle, «Indexing large human-motion databases», presentado en Proceedings of the 30th International Conference on Very Large Data Bases (VLDB'04), 2004, pp. 780-791.

[31] C. A. Ratanamahatana y E. Keogh, «Making Time-series Classification More Accurate Using Learned Constraints», presentado en Proceedings of the 4th SIAM International Conference on Data Mining (SDM'04), 2004, pp. 11-22.

[32] P. S. Bradley y U. M. Fayyad, «Refining Initial Points for K-Means Clustering», presentado en Proceedings of the Fifteenth International Conference on Machine Learning, 1998, pp. 91-99.

[33] Z. Bar-Joseph, G. Gerber, D. K. Gifford, y T. S. Jaakkola, «A New Approach to Analyzing Gene Expression Time Series Data», presentado en Proceedings of the 6th Annual Int©l Conference on Research in Computational Molecular Biology, 2002, pp. 39-48.

[34] M. Matteucci, «Clustering: An Introduction», *A Tutorial on Clustering Algorithms*. [En línea]. Disponible en: http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/index.html. [Accedido: 15-jul-2016].

[35] V. Estivill-Castro, «Why so many clustering algorithms: a position paper», *ACM SIGKDD Explorations Newsletter*, vol. 4, n.º 1, pp. 65-75, 2002.

[36] «Cluster analysis», *Wikipedia, the free encyclopedia*, 02-ago-2016. [En línea]. Disponible en: https://en.wikipedia.org/w/index.php?title=Cluster_analysis&oldid=732730100. [Accedido: 16-jul-2016].

[37] C. D. Manning, P. Raghavan, y H. Schütze, «Single-Link, Complete-Link & Average-Link Clustering», en *Introduction to Information Retrieval*, Cambridge University Press, 2008.

[38] «UPGMA», *Wikipedia, the free encyclopedia*, 03-abr-2016. [En línea]. Disponible en: https://en.wikipedia.org/w/index.php?title=UPGMA&oldid=713319562. [Accedido: 22-ago-2016].

[39] L. Kaufman y P. J. Rousseeuw, «Divisive Analysis (Program DIANA)», en *Finding Groups in Data*, John Wiley & Sons, Inc., 1990, pp. 253-279.

[40] C. D. Manning, P. Raghavan, y H. Schütze, «Divisive clustering», en *Introduction to Information Retrieval*, Cambridge University Press, 2008.

[41] «*k*-means clustering», *Wikipedia, the free encyclopedia*, 05-ago-2016. [En línea]. Disponible en: https://en.wikipedia.org/w/index.php?title=K-means_clustering&oldid=733094958.

[Accedido: 25-jul-2016].

[42] «Lloyd's algorithm», *Wikipedia, the free encyclopedia*, 28-jul-2016. [En línea]. Disponible en: https://en.wikipedia.org/w/index.php?title=Lloyd%27s_algorithm&oldid=731987998. [Accedido: 27-jul-2016].

[43] M. Matteucci, «Clustering - K-means», *A Tutorial on Clustering Algorithms*. [En línea]. Disponible en: http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/kmeans.html. [Accedido: 23-ago-2016].

[44] J. MacQueen, «On convergence of k-means and partitions with minimum average variance», *Annals of Mathematical Statistics*, 1965.

[45] A. Ng, «Choosing the Number of Clusters - Universidad de Stanford», *Coursera*. [En línea]. Disponible en: https://www.coursera.org/learn/machine-learning/lecture/Ks0E9/choosing-the-number-of-clusters. [Accedido: 23-ago-2016].

[46] «Expectation–maximization algorithm», *Wikipedia, the free encyclopedia*, 17-ago-2016. [En línea]. Disponible en: https://en.wikipedia.org/w/index.php?title=Expectation%E2%80%93maximization_algorithm&oldid=734877394. [Accedido: 23-ago-2016].

[47] «DBSCAN», *Wikipedia, the free encyclopedia*, 04-ago-2016. [En línea]. Disponible en: https://en.wikipedia.org/w/index.php?title=DBSCAN&oldid=732913401. [Accedido: 05-jul-2016].

[48] R. J. Hyndman y G. Athanasopoulos, «Forecasting, planning and goals», en *Forecasting: principles and practice*, 2012.

[49] «Time series: Prediction and forecasting», *Wikipedia, the free encyclopedia*, 01-ago-2016. [En línea]. Disponible en: https://en.wikipedia.org/w/index.php?title=Time_series&oldid=732545965. [Accedido: 20-ago-2016].

[50] R. J. Hyndman y G. Athanasopoulos, «Exponential smoothing», en *Forecasting: principles and practice*, 2012.

[51] J. Lin, E. Keogh, S. Lonardi, y B. Chiu, «A symbolic representation of time series, with implications for streaming algorithms», presentado en Proceedings of the 2003 ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, 2003, pp. 2-11.

[52] E. Keogh, K. Chakrabarti, M. Pazzani, y S. Mehrotra, «Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases», *Knowledge and Information Systems*, vol. 3, n.º 3, pp. 263-286.

[53] J. Lin, E. Keogh, L. Wei, y S. Lonardi, «Experiencing SAX: A Novel Symbolic Representation of Time Series», *Data Min. Knowl. Discov.*, vol. 15, n.º 2, pp. 107–144, oct. 2007.

[54] J. Lin, «SAX», 2011. [En línea]. Disponible en: https://cs.gmu.edu/~jessica/sax.htm. [Accedido: 20-ago-2016].

[55] M. Weber, M. Alexa, y W. Müller, «Visualizing time-series on spirals», presentado en Proceedings of the IEEE Symposium on Information Visualization 2001 (INFOVIS'01), 2001,

p. 7.

[56] J. Lin, E. Keogh, S. Lonardi, J. P. Lankford, y D. M. Nystrom, «Visually mining and monitoring massive time series», presentado en KDD ’04 Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, 2004, pp. 460-469.

[57] The DataCamp Team, «Choosing R or Python for data analysis? An infographic», *DataCamp*, 12-may-2015. [En línea]. Disponible en: http://scl.io/Hi4twXG2#gs.CjbudnI. [Accedido: 12-nov-2015].

[58] W. Vorhies, «Programming Python vs R», *William Vorhies’s Blog*, 18-nov-2015. [En línea]. Disponible en: http://www.datasciencecentral.com/profiles/blogs/programming-python-vs-r. [Accedido: 12-dic-2015].

[59] V. Granville, «The Guide to Learning Python for Data Science», *Vincent Granville’s Blog*, 04-may-2016. [En línea]. Disponible en: http://www.datasciencecentral.com/profiles/blogs/the-guide-to-learning-python-for-data-science-2. [Accedido: 18-jul-2016].