



UNIVERSIDAD AUTÓNOMA DE AGUASCALIENTES

CENTRO DE CIENCIAS BÁSICAS

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

TESINA

**ANÁLISIS COMPARATIVO DE ALGORITMOS EN LA CLASIFICACIÓN DE
INTENCIÓN DE MOVIMIENTO MEDIANTE UNA INTERFAZ CEREBRO –
COMPUTADORA ORIENTADO AL USO DE UNA COMPUTADORA.**

Tiempo de redacción: Once meses.

Presenta:

Joel Alejandro Espinoza Sánchez

Para Obtener el Grado de Ingeniería en Computación Inteligente

Directores:

Dra. Aurora Torres Soto

Dra. María Dolores Torres Soto

Comité Evaluador:

Dr. Francisco Javier Álvarez Rodríguez

Dr. Alejandro Padilla Díaz

Aguascalientes, Ags., Mayo 2023.



Aguascalientes, Ags., XX de Mayo de 2023

A quien corresponda:

Por medio de la presente me permito informar que el alumno(a) **Joel Alejandro Espinoza Sánchez** de la carrera de Ingeniería en Computación Inteligente con **ID: 211800**, ha terminado satisfactoriamente su tesina titulada: **“Análisis Comparativo de algoritmos en la clasificación de intención de movimiento mediante una interfaz cerebro – computadora orientado al uso de una computadora.”**, correspondiente a la materia de Seminario de Investigación II.

Para los fines que al interesado convengan.

ATENTAMENTE

Dra. Aurora Torres Soto
Codirector



Aguascalientes, Ags., XX de Mayo de 2023

A quien corresponda:

Por medio de la presente me permito informar que el alumno(a) **Joel Alejandro Espinoza Sánchez** de la carrera de Ingeniería en Computación Inteligente con **ID: 211800**, ha terminado satisfactoriamente su tesina titulada: **“Análisis Comparativo de algoritmos en la clasificación de intención de movimiento mediante una interfaz cerebro – computadora orientado al uso de una computadora.”**, correspondiente a la materia de Seminario de Investigación II.

Para los fines que al interesado convengan.

ATENTAMENTE

Dra. María Dolores Torres Soto
Codirector



Aguascalientes, Ags., XX de Mayo de 2023

A quien corresponda:

Por medio de la presente me permito informar que el alumno(a) **Joel Alejandro Espinoza Sánchez** de la carrera de Ingeniería en Computación Inteligente con **ID: 211800**, ha terminado satisfactoriamente su tesina titulada: **“Análisis Comparativo de algoritmos en la clasificación de intención de movimiento mediante una interfaz cerebro – computadora orientado al uso de una computadora.”**, correspondiente a la materia de Seminario de Investigación II.

Para los fines que al interesado convengan.

ATENTAMENTE

Dr. Francisco Javier Álvarez Rodríguez
Miembro del Comité Tutorial



Aguascalientes, Ags., XX de Mayo de 2023

A quien corresponda:

Por medio de la presente me permito informar que el alumno(a) **Joel Alejandro Espinoza Sánchez** de la carrera de Ingeniería en Computación Inteligente con **ID: 211800**, ha terminado satisfactoriamente su tesina titulada: **“Análisis Comparativo de algoritmos en la clasificación de intención de movimiento mediante una interfaz cerebro – computadora orientado al uso de una computadora.”**, correspondiente a la materia de Seminario de Investigación II.

Para los fines que al interesado convengan.

ATENTAMENTE

Dr. Alejandro Padilla Díaz
Miembro del Comité Tutorial

Resumen

Las interfaces cerebro – computadora permiten un nuevo canal de comunicación directo entre el cerebro y la máquina. Sin embargo para interpretar los pensamientos como el habla imaginada o la intención de movimiento se necesitan herramientas como electroencefalogramas que permitan la recolección de la actividad cerebral. Además, la interpretación de estos datos necesita de algoritmos computacionales de inteligencia artificial que sean efectivos, pero entre toda la gama de algoritmos existentes en este campo se esperaría idealmente comenzar a construir una interfaz cerebro – computadora con algún algoritmo con alta precisión de predicción de intención del movimiento con base en los datos de actividad cerebral de una persona.

Es así que en la presente investigación se planea implementar una interfaz usando una diadema Emotiv Epoch+, el programa elaborado en Python CyKit, OpenViBE y código personal desarrollado en Python para probar experimentalmente cuál de los algoritmos de aprendizaje supervisado entre la máquina de soporte vectorial, el modelo de Naive Bayes y el modelo Random Forest es capaz de predecir con mayor eficiencia intenciones del movimiento con datos obtenidos mediante experimentación.

Palabras clave: Interfaz Cerebro – Computadora. Intención del movimiento. Aprendizaje automático. Máquina de soporte vectorial. Bayes ingenuo. Bosque aleatorio.

Abstract

Brain – computer interfaces provide a new direct communication channel between the brain and the machine. However, to interpret thoughts such as imagined speech or intention of movement, tools such as electroencephalograms are needed to collect brain activity. In addition, the interpretation of this data requires computational algorithms of artificial intelligence that are effective. However, ideally, among the entire range of existing algorithms in this field, it would be expected to start building a brain-computer interface with an algorithm that has high prediction accuracy of movement intention based on a person's brain activity data.

Therefore, the present research aims to implement an interface using an Emotiv EPOC+ headset, the Python CyKit program, OpenViBE, and personal code developed in Python to experimentally test which supervised machine learning algorithm among Support Vector Machine, Naive Bayes, and Random Forest models can predict intention of movement with greater efficiency using data obtained through experimentation.

Key words: Brain – computer interfaces. Intention of movement. Machine learning. Support Vector Machine. Naive Bayes. Random Forest.

Agradecimientos

Sección de agradecimientos.

Si no te caes ¿cómo vas a saber cómo se siente levantarse?
Stephen Curry.

Tabla de contenido

Tabla de contenido -----	1
Índice de figuras -----	4
Índice de tablas -----	7
Índice de ecuaciones -----	8
1. Introducción-----	9
2. Planteamiento del problema-----	13
3. Justificación-----	14
4. Objetivos -----	15
5. Hipótesis-----	16
6. Pregunta de investigación-----	17
7. Marco teórico -----	18
7.1. Introducción a la neuroanatomía-----	18
7.1.1. El cerebro y el sistema nervioso -----	18
7.1.2. El cerebro a nivel macroestructura -----	19
7.1.3. El cerebro a nivel microestructura -----	24
7.1.3.1. Las neuronas-----	24
7.1.3.2. Morfología de las neuronas-----	25
7.1.3.3. Funciones de las neuronas -----	28
7.1.3.4. Interacción entre neuronas -----	33
7.2. Electroencefalografía y estudios del cerebro -----	35
7.2.1. El electroencefalograma -----	35
7.2.2. Breve historia del electroencefalograma -----	36
7.2.3. Ondas encefálicas -----	36
7.2.4. Interfaces cerebro – computadora-----	36

7.2.4.1. El ancestro: interfaces humano – computadora -----	36
7.3. Fundamentos computacionales, inteligencia artificial y aprendizaje automático -----	38
7.3.1. Introducción a la computación -----	38
7.3.1.1. Computación -----	38
7.3.1.2. Programación -----	39
7.3.1.3. El algoritmo -----	40
7.3.2. Inteligencia artificial-----	41
7.3.3. Aprendizaje automático -----	42
7.3.4. Aprendizaje supervisado -----	44
7.3.4.1. La técnica support vector machine -----	47
7.3.4.2. La técnica random forest -----	51
7.3.4.3. La técnica naive Bayes -----	54
7.3.5. Un poco de matemáticas en el preprocesamiento de datos -----	54
7.3.5.1. Normalización z-----	54
7.3.5.2. Transformación de Fourier-----	54
7.4. Los procesos mentales de esta investigación -----	58
7.4.1. Un preámbulo: habla imaginada -----	58
7.4.2. Breviario: evocación de un concepto-----	58
7.4.2. El proceso mental de esta investigación: intención de movimiento -----	58
8. Material y método -----	64
9. Desarrollo -----	65
10. Experimentación y pruebas -----	67
11. Análisis e interpretación de resultados -----	74
12. Conclusiones y discusión-----	78
13. Referencias -----	79
14. Anexos -----	83

14.1. Anexo 1: Configuración para recibir datos desde la diadema hacia el equipo de cómputo usando la paquetería de software de Emotiv, OpenViBE y la librería de Python: Cykit -----	83
14.2. Anexo 2: Código del archivo de Python: functions.py.-----	100
14.3. Anexo 3: Pseudocódigo de la función RunUpdateStream() encontrada en el archivo de Python: functions.py.-----	104
14.4. Anexo 4: Pseudocódigo de la función createDataset() encontrada en el archivo de Python: functions.py.-----	105
14.5. Anexo 5: Acuerdo de privacidad entregado a los usuarios que colaboraron con el experimento.-----	106
14.6. Anexo 6: Protocolo de experimentación.-----	108
14.7. Anexo 7: Evidencias de la experimentación de campo.-----	111

Índice de figuras

7. Marco teórico

7.1. Introducción a la neuroanatomía

Figura 7.1.1. Los cuatro lóbulos cerebrales que pueden observarse desde la corteza cerebral (Sabater, 2020). -----	23
Figura 7.1.2. Representación gráfica de la ubicación del lóbulo insular, señalada en color verde. (Laguna, 2022).-----	26
Figura 7.1.3. Estructuras que conforman el lóbulo límbico. (Triglia, 2016).-----	27
Figura 7.1.4. Morfología de una neurona. (Merck & Co, 2022). -----	30
Figura 7.1.5. Gráfica del potencial de acción de una neurona en función del tiempo. (MDurance, 2021). -----	34
Figura 7.1.6. Gráfica del potencial de acción de una neurona mostrando la interacción iónica. (Olmo, Nave, & Nave, 2022).-----	35

7.2. Electroencefalografía y estudios del cerebro

Figura 7.2.1. Ejemplo de electroencefalograma (Sosa Romano, 2022). -----	39
Figura 7.2.2. Primer registro de un electroencefalograma en un humano (Wikipedia, s.f.). --	41
Figura 7.2.3. Modelo funcional genérico de una BCI (Wikipedia, Interfaz cerebro-computadora, s.f.). -----	45

7.3. Fundamentos computacionales, inteligencia artificial y aprendizaje automático

Figura 7.3.1. Clasificación del machine learning (o aprendizaje automático) en sus tres grandes tipos de aprendizaje (González Barrio, Calleja Ochoa, Gómez-Escudero, Rodríguez Ezquerro, & López de Lacalle Marcaide, 2021).-----	52
Figura 7.3.2. Procedimiento del aprendizaje supervisado (TIBCO Data Science, 2022).-----	54
Figura 7.3.3. Principales algoritmos de machine learning (Betanzos Gómez, 2020).-----	56
Figura 7.3.4. Definición del “margen” entre clases: el criterio que los SVM intentan optimizar (The MathWorks Inc., s.f.).-----	57
Figura 7.3.5. Gráficas de separación del hiperplano de los distintos kernels (Marius, 2020).	59
Figura 7.3.6. Árbol de Decisión aplicado en Economía (Sanabria Castro, 2020). -----	60
Figura 7.3.7. Visualización del “bagging” como separación, usado en Random Forest (Orellana Alvear, 2018). -----	62
Figura 7.3.8. Comparación visual entre el “bagging” y el “boosting” (Sruthi, 2022). -----	63
Figura 7.3.9. Visualización del teorema de Bayes por superposición de dos árboles de decisión (Parrás & Tedesco). -----	64
Figura 7.3.10. El uso del teorema de Bayes en la técnica de naive Bayes (Roman, 2019). -	65
Figura 7.3.11. Gráfica de la función de una distribución normal (Holmes, Illowsky & Dean, 2022). -----	68
Figura 7.3.12. La transformada de Fourier visualmente (Wikipedia, Transformada de Fourier, s.f.).-----	70

Figura 7.3.13. Representación en el plano complejo de un número complejo. Elaboración propia.	72
Figura 7.3.14. Representación de un número complejo bajo el enfoque de magnitud y fase. Elaboración propia.	73
7.4. Los procesos mentales de esta investigación	
Figura 7.4.1. Ubicación de los nodos en la diadema Emotiv usado en su experimentación (Torres-García, Reyes-García, Villaseñor-Pineda, & Ramírez-Cortés, 2013).	81
8. Material y método	
Figura 8.1. Diagrama de pasos a seguir según la metodología planteada.	85
9. Desarrollo	
Figura 9.1. Almohadillas humedecidas en solución salina.	86
Figura 9.2. Diadema lista para su uso.	87
Figura 9.3. Escenario de OpenViBE Designer para experimentación de esta investigación	87
10. Experimentación y pruebas	
Figura 10.1. Ejemplo de uso de la diadema en los voluntarios 1, 8, 7, 9 y 23 de izquierda a derecha, de arriba abajo.	90
Figura 10.2. Resultado de ejecutar la función MousePosition	91
Figura 10.3. Ejemplo de ejecución de la función createDataset con el voluntario 24	93
Figura 10.4. Fragmento del segundo archivo obtenido de la experimentación de campo del voluntario 14.	93
Figura 10.5. Ejecución del algoritmo naive Bayes con el voluntario 17 en su prueba de evocación de un concepto.	94
Figura 10.6. Ejecución de la función de normalización z sobre ambas evaluaciones de los participantes 1 y 2	94
Figura 10.7. Fragmento del segundo archivo obtenido tras la Normalización Z del voluntario 26.	94
Figura 10.8. Ejecución de la función de transformada de Fourier sobre ambas evaluaciones de los participantes 12, 13 y 14	95
Figura 10.9. Ejecución de la división en enfoque magnitud y fase sobre ambas evaluaciones de los participantes 12, 13 y 14	95
Figura 10.10. Fragmento del segundo archivo obtenido tras la transformación de Fourier del voluntario 19.	95
Figura 10.11. Resultado de ejecución del modelo de random forest sobre el segundo archivo del voluntario 10.	96
Figura 10.12. Resultados de ejecución de los tres modelos sobre la primera evaluación de los primeros siete voluntarios.	96
11. Análisis e interpretación de resultados	

Figura 11.1. Comparación de eficiencia de los conjuntos de datos en crudo con ambas evaluaciones -----	97
Figura 11.2. Comparación de eficiencia de los conjuntos de datos normalizados con ambas evaluaciones -----	98
Figura 11.3. Comparación de eficiencia de los conjuntos de datos transformados con ambas evaluaciones -----	99
Figura 11.4. Evolución del SVM en ambas evaluaciones de procesos mentales -----	99
Figura 11.5. Evolución de naive Bayes en ambas evaluaciones de procesos mentales -----	99
Figura 11.6. Evolución de random forest en ambas evaluaciones de procesos mentales-----	99
Figura 11.7. Gráfica de los resultados de ejecución de los tres modelos de machine learning usando los datos recopilados en la experimentación de campo usando la evaluación de evocación de un concepto de los voluntarios. -----	101
Figura 11.8. Gráfica de los resultados de ejecución de los tres modelos de machine learning usando los datos recopilados en la experimentación de campo usando la evaluación de intención de movimiento de los voluntarios. -----	102
Figura 11.9. Gráfica de los resultados de ejecución de los tres modelos de machine learning usando los datos normalizados usando la evaluación de evocación de un concepto de los voluntarios.-----	103
Figura 11.10. Gráfica de los resultados de ejecución de los tres modelos de machine learning usando los datos normalizados usando la evaluación de intención de movimiento de los voluntarios.-----	103
Figura 11.11. Gráfica de los resultados de ejecución de los tres modelos de machine learning usando los datos transformados mediante una transformada de Fourier usando la evaluación de evocación de un concepto de los voluntarios. -----	104
Figura 11.12. Gráfica de los resultados de ejecución de los tres modelos de machine learning usando los datos transformados mediante una transformada de Fourier usando la evaluación de intención de movimiento de los voluntarios.-----	104
14. Anexos	
14.1. Anexo 1: Configuración para recibir datos desde la diadema hacia el equipo de cómputo usando la paquetería de software de Emotiv, OpenViBE y la librería de Python: Cykit	
Figura 14.1.1. Sitio web de Emotiv.-----	115
Figura 14.1.2. Registro para la cuenta Emotiv ID. -----	116
Figura 14.1.3. Emotiv Launcher sin la detección de una diadema conectada al equipo. ---	117
Figura 14.1.4. Diadema Emotiv encendida y dispositivo USB conectado al equipo. -----	117
Figura 14.1.5. Emotiv Launcher detectando una diadema conectada al equipo. -----	118
Figura 14.1.6. Emotiv Launcher guiando al usuario a acomodar la diadema. -----	119
Figura 14.1.7. Sitio web de GitHub donde se encuentra el repositorio del proyecto de CyKit por CymatiCorp. -----	119
Figura 14.1.8. Comando de clonación del repositorio de CyKit-----	120

Figura 14.1.9. Comandos de ejecución de CyKit-----	120
Figura 14.1.10. Ejecución correcta de CyKIT.py usando Python.-----	121
Figura 14.1.11. Explicación de algunas variables de configuración para ejecutar CyKit. ---	121
Figura 14.1.12. Explicación de algunas variables de configuración para ejecutar CyKit. ---	122
Figura 14.1.13. Comando de ejecución personalizada de CyKit -----	123
Figura 14.1.14. CyKit ejecutándose bajo las configuraciones indicadas. -----	123
Figura 14.1.15. Sitio web de OpenViBE. -----	124
Figura 14.1.16. Configuración general de OpenViBE Acquisition Server. -----	124
Figura 14.1.17. Configuración de propiedades de OpenViBE Acquisition Server. -----	125
Figura 14.1.18. Conexión exitosa entre CyKit y OpenViBE Acquisition Server. -----	126
Figura 14.1.19. Ilustración del escenario creado en OpenViBE Designer. -----	127
Figura 14.1.20. Electroencefalograma de OpenViBE Designer en funcionamiento. -----	127
Figura 14.1.21. Tabla de datos de OpenViBE Designer en funcionamiento correcto. -----	128
Figura 14.1.22. Escenario de OpenViBE Designer modificado personalmente.-----	128
Figura 14.1.23. Configuraciones para “CSV File Writer”.-----	129
Figura 14.1.24. Archivo CSV producido tras una ejecución de transmisión de datos desde la diadema. -----	129
Figura 14.1.25. Botón para ciclar la ejecución. -----	130
Figura 14.1.26. Comando de instalación de la librería PyAutoGUI -----	130
Figura 14.1.27. Detección de la posición del mouse con la librería PyAutoGUI-----	130
Figura 14.1.28. Ejecución cíclica del mouse con la librería PyAutoGUI -----	131
Figura 14.1.29. Combinación de los códigos 14.1.27 y 14.1.28 -----	131
14.2. Anexo 2: Código del archivo de Python: functions.py	
Figura 14.2.1. Código del archivo functions.py utilizado en la experimentación-----	142
14.7. Anexo 7: Evidencias de la experimentación de campo.	
Figura 14.7.1. Evidencia de los conjuntos de datos recopilados en el sistema de archivos.151	

Índice de tablas

7. Marco teórico

7.2. Electroencefalografía y estudios del cerebro

Tabla 7.2.1. Clasificación de las ondas encefálicas (Hermann, 1997). ----- 43

7.3. Fundamentos computacionales, inteligencia artificial y aprendizaje automático

Tabla 7.3.1. Cuadro de definiciones de la Inteligencia Artificial presentado por Stuart Russell.
(Russell & Norvig, 2004). ----- 51

14. Anexos

14.7. Anexo 7: Evidencias de la experimentación de campo.

Tabla 14.7.1. Usuarios voluntarios en la experimentación de campo junto a los términos cedidos en el acuerdo. ----- 150

Tabla 14.7.2. Evidencias de los usuarios voluntarios en la experimentación de campo junto al nombre de perfil recibido ----- 152

Índice de ecuaciones

7. Marco teórico

7.3. Fundamentos computacionales, inteligencia artificial y aprendizaje automático

Ecuación 7.3.1. Función de base radial o gaussiana (The MathWorks Inc., s.f.).-----	59
Ecuación 7.3.2. Función lineal (The MathWorks Inc., s.f.). -----	59
Ecuación 7.3.3. Función polinómica (The MathWorks Inc., s.f.).-----	59
Ecuación 7.3.4. Función sigmoide (The MathWorks Inc., s.f.). -----	59
Ecuación 7.3.5. El teorema de Bayes (Parzen, 1987). -----	65
Ecuación 7.3.6. La Regla de Bayes (Walpole, Myers, Myers, & Ye, 2012).-----	66
Ecuación 7.3.7. Función de densidad de probabilidad de una distribución normal (Holmes, Illowsky & Dean, 2022).-----	68
Ecuación 7.3.8. Transformación para obtener las puntuaciones z (Holmes, Illowsky & Dean, 2022). -----	69
Ecuación 7.3.9. La transformada de Fourier (Wikipedia, Transformada de Fourier, s.f.) -----	70
Ecuación 7.3.10. El resultado de la transformada de Fourier simplificado a las dos formas equivalentes de los números complejos (Wikipedia, Transformada de Fourier, s.f.)-----	71
Ecuación 7.3.11. Cálculo de la magnitud de un número complejo. Inferencia propia-----	73
Ecuación 7.3.12. Definición de la función atan2 (Wikipedia, atan2, s.f.)-----	74
Ecuación 7.3.13. Definición de la función atan2 partiendo de la función arcotangente tradicional (Wikipedia, atan2, s.f.)-----	74

7.4. Los procesos mentales de esta investigación

Ecuación 7.4.1. Fórmula del cálculo del CAR (Torres-García, Reyes-García, Villaseñor-Pineda, & Ramírez-Cortés, 2013).-----	79
Ecuación 7.4.2. Fórmula del cálculo de la DWT (Torres-García, Reyes-García, Villaseñor-Pineda, & Ramírez-Cortés, 2013).-----	79
Ecuación 7.4.3. Definición de la base de información disponible D (Torres-García, Reyes-García, Villaseñor-Pineda, & Ramírez-Cortés, 2013).-----	80

1. Introducción

Algunas personas quieren que suceda. Otras desean que suceda. Otras hacen que suceda.
Michael Jordan.

La interacción entre el ser humano y un equipo de cómputo ha mejorado para que esta relación y comunicación sea cada vez más cómoda y deseable para el individuo. La importancia de dar una buena experiencia al usuario es crucial en el mundo actual, donde la tecnología juega un papel fundamental en la vida de las personas. La interacción entre el usuario y la computadora es cada vez más relevante, y una experiencia positiva puede marcar la diferencia entre una persona satisfecha y una insatisfecha.

La mejora de las interfaces humano – computadora es fundamental para garantizar una buena experiencia al usuario. Las interfaces deben ser intuitivas, fáciles de usar y eficientes para garantizar que el usuario no tenga problemas para realizar sus tareas. Una buena interfaz permite que el usuario se sienta cómodo y confiado mientras interactúa con la computadora, lo que a su vez mejora su experiencia y aumenta la satisfacción.

Las interfaces humano – computadora son la clave para una experiencia positiva. Deben ser diseñadas teniendo en cuenta las necesidades y preferencias del usuario, y deben ser continuamente mejoradas y optimizadas para garantizar que el usuario tenga la mejor experiencia posible. La mejora de las interfaces humano – computadora no solo mejora la experiencia del usuario, sino que también aumenta la eficiencia y productividad de la computadora, lo que a su vez aumenta la satisfacción del usuario.

La evolución de la interacción entre humano y computadora ha sido constante desde el surgimiento de las computadoras. Al principio, las interfaces eran limitadas y se basaban en la entrada de comandos a través de teclados y pantallas. Con el tiempo, estas interfaces evolucionaron para incluir gráficos y diseños más atractivos, así como la introducción de dispositivos como el mouse y la pantalla táctil, lo que permitió una interacción más intuitiva y natural.

Sin embargo, con la emergente tecnología de las interfaces cerebro – computadora, estamos presenciando un cambio fundamental en la forma en que interactuamos con las computadoras. Esta tecnología permite a los usuarios controlar las computadoras y otros dispositivos mediante la lectura de señales cerebrales, lo que significa que los usuarios pueden controlar sus dispositivos sin tener que usar dispositivos externos como lo son los tradicionales dispositivos de entrada, teclados o el mouse. La tecnología de las interfaces cerebro – computadora abre nuevas posibilidades para la interacción humano – computadora, lo que puede revolucionar la forma en que las personas interactúan con la tecnología.

Las interfaces cerebro – computadora pueden ser de gran ayuda para personas con ciertas discapacidades que pueden limitar su capacidad para interactuar con la tecnología. Por ejemplo,

para personas con discapacidades motoras o de habla, las interfaces cerebro – computadora pueden ofrecer una forma alternativa y más accesible de controlar las computadoras y otros dispositivos. Esto significa que las personas con discapacidades pueden tener una experiencia más inclusiva y equitativa con la tecnología.

En contraste con la tradicional interfaz humano – computadora, las interfaces cerebro – computadora pueden ser mucho más accesibles para personas con discapacidades. Por ejemplo, las interfaces cerebro – computadora no requieren la capacidad de mover los brazos o las manos, o de hablar, lo que significa que están disponibles para una amplia gama de personas con discapacidades. Igualmente, las interfaces cerebro – computadora son más intuitivas y naturales, como puede observarse de las declaraciones de Kearney (2019) que en su artículo en inglés interpretamos cuestiones como el futuro del diseño de interfaces y experiencia de usuario que evolucionará al diseño de las interfaces cerebro – computadora, buscando crear productos funcionales más disfrutables.

De igual forma, este autor menciona cómo hay aspectos básicos a considerar para la elaboración de estas interfaces como la usabilidad y flexibilidad que se esperaría tener en un molde estándar de interfaz humano – computadora y que trata de trasladarse al campo de las interfaces directamente ligadas con el cerebro, intentando brindar una experiencia más satisfactoria para los usuarios con discapacidades.

La tecnología de las interfaces cerebro – computadora representa un salto tecnológico en la interacción humano – computadora al permitir la traducción directa de pensamientos abstractos a acciones en el mundo real. Esto significa que los usuarios pueden controlar las computadoras y otros dispositivos simplemente pensando en lo que quieren que suceda. Esta capacidad es una característica clave de las interfaces cerebro – computadora y es una de las formas más avanzadas y naturales de interactuar con la tecnología.

El poder operar una computadora mediante la tecnología de las interfaces cerebro – computadora es solo un ejemplo de cómo esta tecnología puede transformar la forma en que interactuamos con el mundo. Al permitir la traducción directa de pensamientos abstractos a acciones concretas, las interfaces cerebro – computadora pueden revolucionar la forma en la que trabajamos, nos comunicamos y experimentamos el mundo que nos rodea. Además, la tecnología de las interfaces cerebro – computadora puede tener un impacto positivo en la vida de las personas con discapacidades y ofrecer nuevas posibilidades para la interacción humano – computadora en el futuro.

La comprensión de los procesos cerebrales es fundamental para el desarrollo de las interfaces cerebro – computadora eficaces y avanzadas. La neurociencia y la neuroanatomía son disciplinas clave que han contribuido en gran medida a nuestra comprensión del cerebro y cómo funciona. Al

entender los procesos cerebrales, los investigadores pueden desarrollar tecnologías que permitan la comunicación directa entre el cerebro y la computadora.

Los estudios de neurociencia y neuroanatomía han permitido avances en la comprensión de la estructura y función del cerebro y cómo se relacionan con los procesos cognitivos y el comportamiento. Esta comprensión es crucial para el desarrollo de interfaces cerebro – computadora avanzadas que puedan interpretar los patrones cerebrales y traducirlos en acciones concretas en el mundo real. Asimismo, la neurociencia y la neuroanatomía también han contribuido a la comprensión de las diferencias individuales en el cerebro y cómo éstas pueden afectar la interacción con la tecnología.

Al comprender cómo funciona el cerebro, los investigadores pueden desarrollar tecnologías que permitan la comunicación directa entre el cerebro y la computadora y así mejorar la interacción humano – computadora. Además, la neurociencia y la neuroanatomía continúan evolucionando y desarrollándose, lo que significa que las interfaces cerebro – computadora seguirán mejorando y evolucionando a medida que los investigadores aprendan más sobre el cerebro y cómo funciona. Es decir, tanto la neurociencia como la neuroanatomía son esenciales para el desarrollo de interfaces cerebro – computadora avanzadas y eficaces y para la mejora continua de tecnologías que permiten la interacción humano – computadora.

La invención de los electroencefalogramas (EEG) ha tenido un impacto significativo en la forma en que se diagnostican y tratan las enfermedades del cerebro. Como Palucci y otros (2023) describen entre los usos médicos de los EEG – así como se verá más adelante – un EEG es una técnica no invasiva que registra la actividad eléctrica del cerebro y permite a los médicos ver los patrones de actividad cerebral. Esto les permite diagnosticar enfermedades del cerebro como el trastorno del espectro autista, la epilepsia y otras enfermedades neurológicas.

La capacidad de leer la actividad cerebral con un EEG ha permitido una mejor comprensión de las enfermedades del cerebro y ha ayudado a desarrollar tratamientos más eficaces. Palucci y otros (2023) describen cómo estas tecnologías ayudan en la rehabilitación de personas “con desórdenes neurológicos como la epilepsia, el trastorno por déficit de atención o hiperactividad o la comunicación con personas con parálisis cerebral” (Palucci, y otros, 2023).

Los EEG pueden ayudar a identificar la presencia de anomalías en la actividad cerebral que están asociadas con ciertas enfermedades. Además, los EEG también pueden ayudar a los médicos a monitorear la respuesta al tratamiento y ajustarlo en consecuencia. Es así que la invención de los EEG ha sido fundamental para el diagnóstico de enfermedades del cerebro y ha ayudado a mejorar el tratamiento de estas enfermedades.

De hecho, un EEG puede ser utilizado como una pieza clave de una interfaz cerebro – computadora, ya que permite a los investigadores y desarrolladores capturar la actividad cerebral en tiempo real y

convertirla en señales electrónicas que pueden ser procesadas y utilizadas por una computadora. Al registrar la actividad cerebral, los EEG pueden ayudar a identificar patrones específicos de actividad que pueden ser utilizados para controlar dispositivos externos, como robots o dispositivos de asistencia.

Por otra parte, los EEG también pueden ser utilizados para crear una experiencia de interacción más intuitiva y natural para el usuario, permitiéndole controlar dispositivos y aplicaciones simplemente con sus pensamientos. La capacidad de leer la actividad cerebral a través de un EEG permite a los desarrolladores crear interfaces cerebro – computadora más precisas y confiables, lo que puede tener un impacto significativo en la forma en que las personas interactúan con la tecnología.

La implementación de interfaces cerebro – computadora ha sido un desafío debido a la complejidad de predecir lo que el usuario realmente desea con sólo un impulso cerebral. Debido a la naturaleza subjetiva y variada de las percepciones humanas, es difícil traducir la actividad cerebral en una acción específica y predecible para la computadora.

Sin embargo, esta dificultad ha sido abordada a través del uso del aprendizaje automático, una tecnología que permite a la computadora aprender a partir de la experiencia y mejorar con el tiempo. Al usar técnicas de aprendizaje automático, las interfaces cerebro – computadora pueden aprender a identificar patrones específicos de actividad cerebral y traducirlos en acciones en tiempo real, haciendo la interacción más efectiva y eficiente.

El aprendizaje automático se ha utilizado para mejorar la precisión y confiabilidad de las interfaces cerebro – computadora, al mismo tiempo que permite a los desarrolladores tener un mejor entendimiento de las preferencias y comportamientos de los usuarios. Además, al ser una tecnología en constante evolución, el aprendizaje automático puede ser una herramienta clave en el desarrollo de nuevas aplicaciones y tecnologías basadas en interfaces cerebro – computadora.

Es así que la implementación de interfaces cerebro – computadora ha sido un desafío debido a la complejidad de predecir lo que el usuario realmente desea con sólo un impulso cerebral. Sin embargo, el uso del aprendizaje automático ha permitido abordar esta problemática y mejorar la precisión y confiabilidad de las interfaces cerebro – computadora. Con el tiempo, se espera que esta tecnología continúe evolucionando y permita nuevas aplicaciones y tecnologías en el futuro.

El habla imaginada es un fenómeno neurológico en el que el cerebro genera señales eléctricas similares a las producidas durante la producción de habla. Un fenómeno similar es la intención de movimiento, el cual consiste en la voluntad de una persona de realizar alguna acción en específico, labor que comienza en el cerebro y que se encuentra estrechamente relacionada con el habla imaginada, incluso implicando las mismas zonas cerebrales para estas operaciones.

Gracias al aprendizaje automático, se está haciendo un acercamiento cada vez más preciso en la interpretación de estas ondas cerebrales como habla imaginada o intención de movimiento. Esto significa que cada vez es más posible comprender lo que el usuario realmente desea con sólo su impulso cerebral, lo que sería un gran avance en el desarrollo e implementación de las interfaces cerebro – computadora.

Esto quiere decir que mediante modelos de aprendizaje automático es posible realizar estas interpretaciones, gracias a investigaciones que más adelante se abordarán y concluyen en resultados muy esperanzadores, sin embargo, entre la gran variedad de modelos de aprendizaje automático ¿cuál puede ser el más efectivo?

En la presente investigación se tratará de responder esta pregunta comparando tres modelos de aprendizaje automático, abordando a mayor profundidad este campo y explorando tres interesantes técnicas dentro de esta área, así como el comportamiento cerebral a mayor detalle, igualmente se explorará el EEG y finalmente se hablará de la intención de movimiento junto con los estudios recientes del habla imaginada dentro de este tema del conocimiento.

En los capítulos sucesivos se expone el tratamiento de este problema realizando su respectivo planteamiento en el capítulo dos y la justificación de la presente investigación en el capítulo tres. Los objetivos se establecen claros en el capítulo cuatro mientras que en los capítulos cinco y seis se menciona la hipótesis y la pregunta de esta investigación.

El capítulo siete brinda todo el marco teórico requerido de esta investigación dividido en subapartados. El apartado 7.1 trata todos los conceptos referentes a neuroanatomía sobre el cerebro que deben de cubrirse para entender el objeto de estudio que es el cerebro y la forma en la que realiza sus funciones. En el apartado 7.2 se introduce un poco de electrónica, hablando de lo que es un electroencefalograma y cómo esta herramienta realiza estudios enfocados en el cerebro.

El apartado 7.3 aclara toda la información de ciencias de la computación necesaria, desde una introducción hasta la cuestión técnica de los algoritmos de aprendizaje supervisado que se usarán en este proyecto. El apartado 7.4 salta finalmente uniendo todo esto para que el lector sepa qué problema se quiere atacar con todo el conocimiento brindado hasta ese punto, siendo altamente importante la cuestión de procesos cerebrales en este grupo.

Después el capítulo ocho describe los materiales y el método que se necesitan y que se proponen para el desarrollo de la experimentación, la cual se expone en el capítulo nueve. En el capítulo diez, la experimentación se lleva a un terreno práctico mostrando evidencias realizadas durante las pruebas para interpretar los resultados obtenidos en el capítulo once.

Finalmente, en el capítulo 12 se cierra la presente exploración dando una evaluación de los aspectos logrados gracias a esta investigación.

2. Planteamiento del problema

No podemos resolver problemas pensando de la misma manera que cuando los creamos.
Albert Einstein.

Pese a los avances de comodidad y usabilidad dentro de los entornos de interacción entre personas y computadoras, aún existen muchas barreras en el uso de los equipos de cómputo, especialmente para personas con ciertas dificultades o discapacidades; cuestiones como la coordinación motriz en el movimiento de las manos para escribir en el teclado o mover el ratón que pueden ser un aspecto cotidiano si no se lidia con alguna discapacidad.

Por otro lado, las personas que afrontan estas discapacidades que les impiden el uso de los métodos de entrada tradicionales como el teclado o el ratón ven limitada la forma de comunicarse con la computadora. Es por ello que, para tratar de brindar mayor inclusión y accesibilidad a un sector del público, las interfaces cerebro – computadora traen consigo esa posibilidad de eliminar esta barrera.

Basta con prestar atención a las declaraciones de Marte (2019) quien se mantiene entusiasta creyendo que esta tecnología puede ayudar a un público general proporcionando experiencias más motivadoras y facilitando muchas actividades que requieren del uso de un equipo de cómputo; es así que diseñar una interfaz cerebro – computadora reduciría este problema que impide a ciertos individuos tener una experiencia satisfactoria al hacer uso de un equipo de cómputo.

Sin embargo, también se debe contemplar elaborar una interfaz de este tipo eficiente y funcional. Desde los enfoques abordados por investigadores como el caso de Torres-García, Reyes-García, Villaseñor-Pineda, y Ramírez-Cortés (2013) en el que realizan un análisis de señales electroencefalográficas, se puede apreciar que implementar este tipo de interfaces también significa enlazarlo con el uso de algoritmos de machine learning, pues estos tienen el papel de interpretar estas señales y convertirlas en las acciones que se desean hacer haciendo uso del equipo de cómputo.

Es por ello que, al plantearse realizar una interfaz y encontrarse con la gama de algoritmos existentes, es común darse cuenta que la cantidad de técnicas de machine learning para analizar estos datos es muy extensa. Pueden reducirse los candidatos iniciales a una selecta muestra de algoritmos como lo son los de aprendizaje supervisado, pero dentro de estos algoritmos, la variedad sigue siendo vasta.

Tomando en cuenta la extensa variedad de algoritmos de aprendizaje automático, así como la complejidad de análisis de las señales de actividad cerebral, en este trabajo se han elegido tres técnicas de aprendizaje supervisado: support vector machine, random forest y naive Bayes, y así dilucidar la problemática que radica en conocer cuál de estos tres algoritmos realizaría una clasificación de intención de movimiento con mayor eficiencia a partir de las señales obtenidas de un EEG y transformarlos en una intención deseada de accionar un método de entrada de un equipo

de cómputo. Es así que, para la presente investigación se tomará un conjunto de acciones reducido como las distintas intenciones posibles para la experimentación aquí planteada.

Así la problemática descrita es del tipo comparativa, buscando de entre los tres modelos previamente mencionados, aquel que tenga una mejor adaptabilidad para realizar una relación entre los valores que un EEG proporcione al algoritmo y un selecto grupo de palabras y acciones definidas que permitan hacer una evaluación a pequeña escala de intención de movimiento.

Con estas bases planteadas, la investigación se centrará en encontrar el algoritmo de mejor precisión que realice una clasificación más eficiente de las señales electroencefalográficas obtenidas por un EEG de usuarios voluntarios para recopilar datos de sus respectivas señales de actividad cerebral para su posterior etiquetado bajo un conjunto reducido de intenciones de movimiento orientadas a operar un equipo de cómputo y finalmente usar estos conjuntos de datos como evaluación de estos algoritmos; finalmente comparar la eficiencia de cada uno de ellos para así, concluir cuál algoritmo es el más eficiente en la clasificación de intención de movimiento a partir de las señales obtenidas de un EEG.

3. Justificación

La inteligencia consiste no sólo en el conocimiento, sino también en la destreza de aplicar los conocimientos en la práctica.
Aristóteles.

La Organización Mundial de la Salud (OMS) en un artículo del 2023, estima que alrededor del 16% de la población global actualmente experimentan alguna discapacidad significativa. La organización declara que este número está en aumento debido en parte al envejecimiento de la población y a la prevalencia de enfermedades no transmisibles, lo cual es un sector de la población numeroso.

Por otra parte, la oficina de comunicaciones del NIH (2019) declaran que el uso de la tecnología “permite que los estudiantes con discapacidades compensen ciertas deficiencias. Esta tecnología especializada promueve la independencia y disminuye la necesidad de otro apoyo”, referente a estudios con estudiantes en centros de educación.

Los mismos autores del NIH (2019) también señalan que, al ayudarse de la tecnología, estos equipos permiten que las personas trabajen de mejor manera, aprendan en entornos típicos a nivel educacional, accedan a la información a través de las computadoras y la lectura, disfruten de la música, deportes y artes y, por consiguiente, les permita a las personas participar plenamente en una vida comunitaria.

Esta investigación cimenta su justificación – desde un punto de vista social – en la mejora de la interacción que existe entre el ser humano y la computadora, tratando de dar un paso más cerca en la dirección del uso de las interfaces cerebro – computadora como una vía más directa en la comunicación de los individuos y la operación por medio del cerebro, en este caso, de un equipo de cómputo para usuarios con discapacidades, orientado a que este mismo sector de la población tenga una herramienta que les haga más accesible la comunicación con un equipo de cómputo.

Hay personas que por dificultades motrices les cuesta hacer un manejo óptimo de un equipo de cómputo. Desarrollando una interfaz cerebro – computadora que funcione adecuadamente podría ayudar a este sector de la población a tener una mejoría en el control de una computadora.

Es así también que, se espera que sea un paso más para permitir una mayor inclusión en el uso de los dispositivos computacionales para personas con ciertas discapacidades, que no les permitan usar de forma óptima métodos de entrada tradicionales como el teclado y el mouse; así, este tipo de usuarios podría operar el equipo por vías que les sean accesibles.

Por otra parte, desde un enfoque computacional, la justificación de esta investigación yace en analizar algunos algoritmos de aprendizaje supervisado con el objetivo de ponerlos a prueba y verificar cuál de ellos es el que mejor realiza la clasificación de intención de movimiento dentro de un conjunto de acciones reducidas enfocadas al manejo de un sistema computacional.

Cabe aclarar que esta investigación trata de ser una piedra más en el camino para que el futuro de la investigación por esta vía siga acomodando el camino y que la justificación de la presente en trabajo futuros siga reforzándose con mayor exploración en esta área y mayores avances que guíen por la senda que se matiza en este apartado de la presente investigación.

4. Objetivos

No hay ningún viento favorable para el que no sabe a qué puerto se dirige.
Arthur Schopenhauer.

El objetivo general es:

- Comparar y encontrar el modelo de aprendizaje supervisado, entre el support vector machine, el modelo random forest y el modelo naive Bayes, cuál clasifica con mayor eficiencia datos obtenidos a partir de una diadema Emotiv EPOC+ orientado a la intención de movimiento usando un conjunto de acciones dados a los usuarios de prueba.

Así también, se establecieron los siguientes objetivos específicos:

- I. Realizar una conexión eficiente entre el dispositivo Emotiv EPOC+ y un equipo computacional que permita extraer los datos de la actividad neuronal de un usuario.
- II. Adaptar las herramientas necesarias, tales como el programa desarrollado en Python por CymatiCorp: CyKit, el programa OpenViBE y el software de Emotiv, para que la recepción de datos del dispositivo que actúa como EEG sea realizada correctamente.
- III. Realizar un preprocesamiento de los datos obtenidos en la experimentación de campo con los usuarios que colaboraron con registrar su actividad cerebral, pasando estos conjuntos de datos por técnicas de normalización y transformación que ayuden en una mejora del rendimiento de los modelos de machine learning.
- IV. Implementar los algoritmos support vector machine, random forest y naive Bayes para hacer las pruebas de aprendizaje sobre los datos de actividad neuronal.
- V. Analizar cuál de los modelos previamente mencionados clasifica con mayor eficiencia la actividad neuronal del usuario en intención del movimiento que permitan operar una computadora.

Estos son los objetivos que fungen como brújula del proyecto en cuestión.

5. Hipótesis

Creer posible algo es hacerlo cierto.
Friedrich Hebbel.

Como hipótesis, los modelos de aprendizaje supervisado tendrán un desempeño decente en la clasificación de intención de movimiento orientado al control de un equipo de cómputo mediante una interfaz cerebro – computadora, sin embargo considero que el modelo que mostrará una mayor eficiencia será la support vector machine mostrando resultados de predicción notablemente mejores que los otros dos competidores en este campo. El segundo lugar lo tendrá el algoritmo de random forest y el que demuestre un peor rendimiento – aunque aún decente – será naive Bayes.

6. Pregunta de investigación

A un gran corazón, ninguna ingratitud lo cierra, ninguna indiferencia lo cansa.
Leon Tolstoi.

La pregunta de investigación la cual es el pilar principal de esta investigación es la siguiente:

- ¿Cuál modelo de aprendizaje supervisado entre el support vector machine, random forest y naive Bayes es el más eficiente para la clasificación de intención de movimiento orientado al control de un equipo de cómputo mediante una interfaz cerebro – computadora?

7. Marco teórico

7.1. Introducción a la neuroanatomía

Soy un cerebro, Watson. El resto de mí es un mero apéndice.
Arthur Conan Doyle: La piedra de Mazarino.

7.1.1. El cerebro y el sistema nervioso

El cerebro es un órgano muy interesante, que ha cautivado la atención de muchos investigadores y la curiosidad de las personas, pues como lo señala la corporación Caldaria en su blog HDOSO Magazine, “el cerebro es el órgano más complejo del cuerpo, al menos en los vertebrados, y lo es porque es este órgano el que controla el resto de órganos del cuerpo y el que, en definitiva, nos define como seres humanos” (Caldaria, 2020). Nazareno, estudiante de la Universidad de Harvard mientras escribía un artículo expresando su fascinación por el cerebro declara cómo a su percepción, este órgano es fascinante y los hallazgos sobre éste siguen asombrando a la comunidad científica enfocada en esta parte del cuerpo (Nazareno, 2020).

Hill define al cerebro desde una perspectiva etimológica “del latín *cerebrum*, con su raíz indoeuropea *ker*, cabeza, en lo alto de la cabeza y *brum*, ‘llevar’; teniendo el significado de ‘lo que lleva la cabeza’. Es un órgano que centraliza la actividad del sistema nervioso y existe en la mayor parte de los animales” (Hill, 2006). Este órgano “es el más complejo del cuerpo. El telencéfalo adquiere su máximo desarrollo y está formado por los hemisferios cerebrales. El cerebro humano contiene en la corteza cerebral, un número estimado de 20,000,000,000 (20 mil millones, 2×10^{10}) de neuronas” (von Bartheld, Bahney, & Herculano-Houzel, 2016), (Pelvig, Pakkenberg, Stark, & Pakkenberg, 2008), (Herculano-Houzel, 2009).

Puede deducirse gracias a las definiciones de los autores dadas previamente que el cerebro humano es el órgano central que recibe todo impulso nervioso que el cuerpo detecta y que tanto por su estructura como por su funcionamiento, es necesario dividirlo en algunas clasificaciones, como lo realizan Latarjet y Ruiz Liard, pues ellos señalan que está formado “por las estructuras derivadas del Telencéfalo y el Diencéfalo, los dos sectores anteriores del Prosencéfalo embrionario. Ocupa el sector anterior y superior del cráneo llamados fosa craneal anterior y fosa craneal media” (Latarjet & Ruiz Liard, 2004).

Al ser la piedra angular del sistema nervioso, cabe definir a este grupo como “el conjunto de órganos que regulan, coordinan e integran todas las actividades del organismo. Asimismo, constituye una unidad funcional compleja que se puede dividir, desde el punto de vista didáctico, en dos componentes morfológicos fundamentales: el sistema nervioso central (SNC) y el sistema nervioso periférico (SNP)” (Ojeda Sahagún & Icardo de la Escalera, 2004).

Ojeda Sahagún e Icardo de la Escalera (2004) también señalan en su obra algunas otras consideraciones entre esta división, pues “el SNC agrupa todas las estructuras del sistema nervioso que se encuentran alojadas dentro del estuche osteofibroso formado por la cavidad craneal y el conducto vertebral. Por situarse en la línea media, a veces se denomina neuroeje” (Ojeda Sahagún, 2004). Por otra parte, el SNP, comentan Ojeda Sahagún e Icardo de la Escalera (2004), “comprende el resto de estructuras nerviosas que, aunque en su origen siguen un breve trayecto dentro de la cavidad craneal o del conducto vertebral, se sitúan fuera del estuche osteofibroso”.

El SNC se divide en encéfalo y médula espinal según Ojeda Sahagún (2004). Para efectos de la presente investigación, se centrará la atención en esta parte del sistema nervioso, específicamente en el encéfalo, que a su vez está constituido por el tronco del encéfalo, el cerebelo, el diencéfalo y los hemisferios cerebrales. “El conjunto del diencéfalo y los hemisferios cerebrales se denomina cerebro” (Ojeda Sahagún & Icardo de la Escalera, 2004) y se enfocará la atención en este grupo.

Con lo anterior mencionado, puede realizarse la siguiente definición funcional para efectos de esta investigación, pues el cerebro abarca muchos conceptos gracias a su alta complejidad, pero también los estudios analizados coinciden con que éste es el punto central del sistema nervioso y es nuestro principal procesador de información, el que recibe los estímulos nerviosos del exterior y los convierte en conocimiento a través de la percepción, la razón, las emociones y las otras formas de comprensión de nuestro entorno. Es capaz también de controlar los movimientos voluntarios, el habla y nuestro productor de inteligencia.

Aunque se sabe que el cerebro es capaz de realizar estas acciones, es también importante saber el cómo es capaz de realizar estas actividades. Para ello requerimos de analizar la estructura del órgano tanto a nivel general como a un nivel específico.

7.1.2. El cerebro a nivel macroestructura

A un nivel de análisis amplio, Kandel, Schwartz y Jessel (2000) mencionan en su obra que el cerebro se divide en dos hemisferios cerebrales denominados hemisferio izquierdo y derecho, los cuales “son aproximadamente simétricos, sin embargo el izquierdo es ligeramente mayor. Están separados por la profunda cisura medial. Están cubiertos por una capa cortical sinuosa, la corteza cerebral, formada por sustancia gris” (Kandel, Schwartz, & Jessel, 2000).

Anatomistas como los anteriormente mencionados clasifican cada hemisferio en seis lóbulos: lóbulo frontal, lóbulo parietal, lóbulo occipital, lóbulo temporal, lóbulo insular y lóbulo límbico. Huang (2021) también divide al cerebro de esta manera mencionando que “el cerebro está dividido por una fisura longitudinal en dos hemisferios, cada uno compuesto por seis lóbulos distintos” indicando los previamente citados.

Sin embargo, autores como Triglia (2015) difieren de la clasificación previamente acordada. Esto porque, como se percibe en su artículo, él menciona una categorización en cinco lóbulos, siendo los lóbulos frontal, parietal, occipital y temporal aquellos que coinciden con el orden anterior; por otra parte, Triglia trata el lóbulo insular como un aspecto propio dándole el nombre de ínsula y definiéndolo como “una parte de la corteza que queda oculta entre el resto de lóbulos del cerebro y, para verla, es necesario apartar entre sí los lóbulos temporal y parietal. Es por eso que frecuentemente no es tenida en cuenta como un lóbulo más” (Triglia, Los 5 lóbulos del cerebro y sus distintas funciones, 2015).

A diferencia de la organización anteriormente mencionada, puede observarse que el lóbulo que no se menciona es el lóbulo límbico; esto se debe a que Triglia (2015) lo considera como todo un sistema y no como un lóbulo parte del cerebro. Él explica que es, de hecho la ínsula, la que mayor conexión tiene con éste, el sistema límbico y que quizá es la encargada de mediar entre este sistema y los procesos cognitivos que se realizan en la central de control del ser humano.

Para efectos de la presente investigación, se tomará como base la división que proponen autores como Huang (2021) separando el cerebro en seis agrupaciones distintas, pues, pese a reconocer su complejidad al comprenderse como sistema, en esta exploración el objetivo busca más la forma en la que estas partes del cerebro trabajan conjuntamente entre sí, por lo que se considera indiferente tratarlo como lóbulo límbico o como sistema límbico y aunque se le nombre a continuación bajo cualquiera de los dos nombres anteriormente aludidos, la intención es hacer mención a este conjunto de estructuras del cerebro.

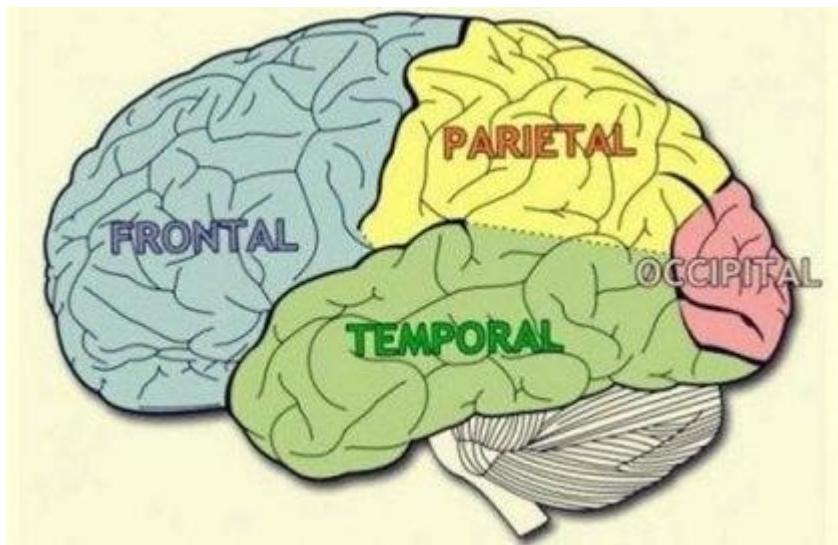


Figura 7.1.1. Los cuatro lóbulos cerebrales que pueden observarse desde la corteza cerebral. (Sabater, 2020).

En la figura 7.1.1 puede observarse el cerebro y la división de los lóbulos que son visibles desde la corteza cerebral. La división de esta forma del cerebro permite centrarse en las diferentes

secciones sabiendo el hecho de que “cada lóbulo cerebral presente una serie de características propias no significa que cada estructura controle casi en “exclusiva” una determinada tarea. Muchas actividades y procesos se superponen a través de las distintas regiones cerebrales” (Sabater, 2020).

Comenzando por analizar los lóbulos frontales, éstos poseen grandes muestras de la evolución humana, pues “situados en la parte frontal de la cabeza, y justo debajo de los huesos frontales del cráneo y cerca de la frente, conforman la región más afinada de nuestro cerebro, la que más tiempo tardó en evolucionar y aparecer” (Sabater, 2020).

Otros autores resaltan la importancia de este lóbulo “para las funciones cognitivas y el control de la actividad o el movimiento voluntario” (Personal Mayo Clinic, 2021). Otros autores como Kolb y Whishaw (2014) resaltan tres áreas funcionales que comprende este lóbulo siendo éstas las siguientes:

1. El área motora, que proyecta el movimiento que deben realizar las extremidades y los movimientos faciales.
2. El área premotora, la cual influye en la ejecución del movimiento.
3. El área prefrontal, que controla procesos cognitivos de modo que los comportamientos, movimientos y conductas sean los apropiados.

Sabater (2020) también menciona que las tareas más notorias que este lóbulo realiza son las siguientes:

- La producción de habla y lenguaje.
- Procesos cognitivos que nos permiten planificar, fijar la atención, memorizar datos a largo plazo, comprender lo que vemos, regular nuestras emociones, etc.
- Comprender y reaccionar ante los sentimientos de los demás, como por ejemplo, la empatía.
- Regulación de la motivación y búsqueda de recompensas.

El siguiente lóbulo se trata del lóbulo parietal. Existen redactores que dividen el lóbulo en cuestión en dos fragmentos, “uno implica la sensación y la opinión y la otra se refiere a integrar la entrada sensorial, sobre todo con el sistema visual. La primera función integra información sensorial para formar un solo percepción (cognición). La segunda función construye un sistema coordinado espacial para representar el mundo alrededor de nosotros” (Kandel, Schwartz, & Jessel, 2000).

La definición de esta parte del cerebro coincide con las que escritores como Sabater (2020) señalan, como su importante papel en la percepción de los sentidos, en el razonamiento espacial, el movimiento de nuestro cuerpo y nuestra orientación, pues tal como lo dice esta autora, es en esta área donde “se capta la información sensorial relativa a la mayoría de nuestros órganos

sensoriales. Es aquí donde se procesa y regula la sensación del dolor, la presión física y la temperatura, etc.” (Sabater, 2020). Aunque también esta autora remarca que gracias al área parietal, también es posible comprender la naturaleza de los números y la relación que encontramos con las matemáticas.

El siguiente lóbulo es el lóbulo occipital. Este es el lóbulo más pequeño y aunque Sabater (2020) lo considera un camino de paso de los demás procesos, también señala que este lóbulo sí realiza algunos procesos importantes como los siguientes:

- Participa en los procesos de percepción y reconocimiento visual.
- Tiene alta importancia en todo lo relativo al sentido de visión, ya que su corteza integra diversas áreas visuales como la detección de patrones, procesamiento de información y envío de esta información a otras áreas del encéfalo.
- Ayuda en la diferenciación de colores.
- Participa también en la elaboración de las emociones y pensamientos.

De hecho, Huang (2021) refuerza esta idea al mencionarnos en su artículo dedicado a las lesiones en los distintos lóbulos cerebrales que el lóbulo occipital es el centro principal de procesamiento de la información visual pues interpreta la visión, forma recuerdos visuales e integra las percepciones visuales con información espacial la cual recolecta por el lóbulo parietal adyacente.

Dentro de este conjunto tenemos también al lóbulo temporal. Esta parte del cerebro, según Sabater (2020) se encarga especialmente de las siguientes tareas:

- Nos ayuda a reconocer los rostros.
- Tiene estrecha relación con la articulación del lenguaje, la comprensión de sonidos, voces y la música.
- Facilita el equilibrio.
- Participa en la regulación de emociones como la motivación, la rabia, la ansiedad, el placer, entre otras.

Sin embargo, la autora también destaca que es “muy complicado asociar a cada una de estas estructuras a una única función especializada. Todas dependen unas de otras, todas se hallan conectadas y favorecen esa armonía perfecta” (Sabater, 2020), de modo que todos los lóbulos se desempeñen juntos en consonancia.

Sabater menciona que, gracias a los estudios del cerebro realizados, se ha hecho el hallazgo de la quinta región: el lóbulo insular o la ínsula lobular; descrita por esta autora como “un lóbulo oculto justo debajo de los lóbulos temporal, frontal y parietal. Es un área muy recóndita y de complejo acceso localizada entre los vasos venosos y arterias” (Sabater, 2020). Puede observarse una muestra gráfica de su ubicación en la figura 7.1.2.

Sabater menciona que, pese a que no se sabe a ciencia cierta cuáles son sus funciones, gracias a otros procesos y correlaciones, el lóbulo insular participa en el sentido del gusto, el control visceral, la somatopercepción y otros procesos emocionales de los cuales forma parte junto con el sistema límbico. Otros autores comparten este hecho de la quinta región, pues como menciona Laguna, “el lóbulo de la ínsula es el que menos se ha estudiado. Por esta razón, es considerado un lóbulo nuevo para los estudiantes de ciencias de la salud ya que hay muy poca información sobre esta estructura” (Laguna, 2022).

Además, Laguna menciona otras funciones de esta área del cerebro, encontradas gracias a que “diversos estudios de neuroimagen han relacionado al lóbulo de la ínsula con los deseos, los antojos y las adicciones. Asimismo, se ha evidenciado que este lóbulo juega un papel de suma importancia en cuanto a los trastornos psiquiátricos, tales como la esquizofrenia, los trastornos de pánico, el estrés postraumático y el trastorno obsesivo-compulsivo” (Laguna, 2022).

Redactado bajo el nombre de la ínsula, Martínez (2022) relata que esta región del cerebro “se encuentra implicada en múltiples funciones: parece tener una gran relación con las emociones básicas como el amor, la tristeza, el odio o la felicidad. Además, tiene un papel muy importante en cuanto a la regulación del cuerpo para conseguir la homeostasis y su implicación en la percepción de conciencia de nosotros mismos y de las experiencias emocionales subjetivas, es otro de sus papeles más destacados” (Martínez, 2022).

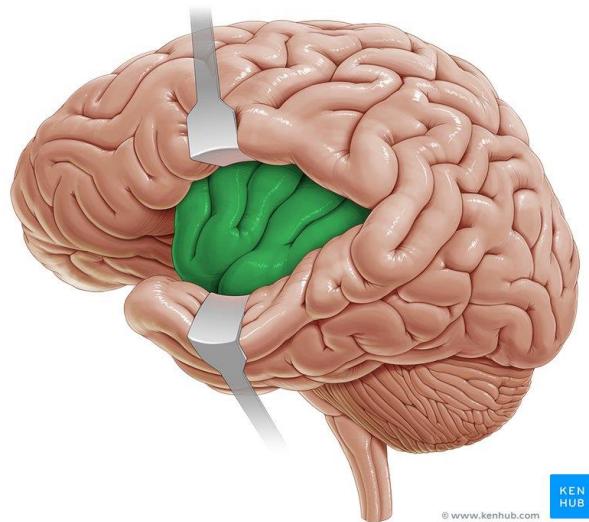


Figura 7.1.2. Representación gráfica de la ubicación del lóbulo insular, señalada en color verde. (Laguna, 2022).

Finalmente se tiene el sexto lóbulo: el lóbulo límbico. Éste “comprende una serie de estructuras situadas en la zona medial de los hemisferios cerebrales. Interviene principalmente en la expresión de afectos y en la memoria” (Interpsiquis, 2022). Estos autores mencionan las estructuras que forman parte de este lóbulo, mismas que pueden observarse en la figura 7.1.3, siendo las siguientes:

- Estructuras corticales: cingulado y gyrus parahipocámpico.
- Formaciones hipocámicas: gyrus dentado, hipocampo y complejo subicular.
- La amígdala.
- El núcleo *accumbens*.
- El hipotálamo.
- El tálamo (núcleo anterior y núcleo dorsomedial).
- Otras estructuras corticales como el orbitofrontal y el polo temporal.

Sistema Límbico

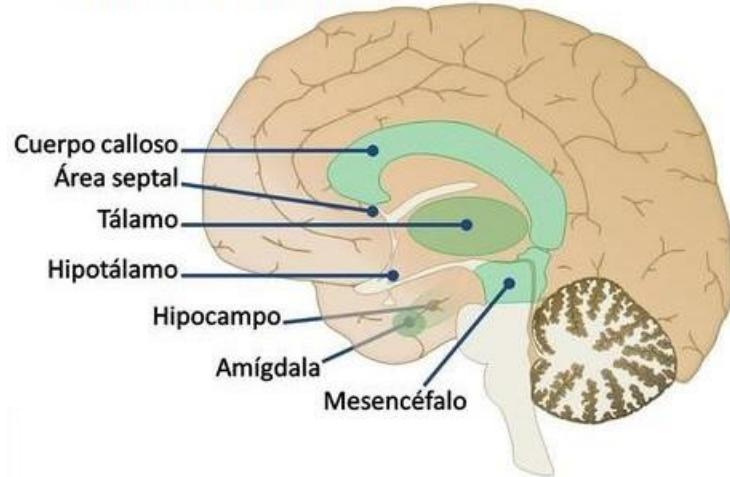


Figura 7.1.3. Estructuras que conforman el lóbulo límbico. (Triglia, Sistema límbico: la parte emocional del cerebro, 2016).

Con base en las definiciones y proposiciones de los autores previos, es posible enumerar las principales actividades de cada uno de los lóbulos cerebrales; así, serían las siguientes:

- **Lóbulo frontal:** Trabaja principalmente en el habla y el lenguaje, ayuda en fijar la atención y la memoria a largo plazo. Regula la motivación y búsqueda de recompensas, así como la empatía.
- **Lóbulo parietal:** Tiene alta importancia en la percepción de los sentidos, razonamiento espacial, percepción de nuestro cuerpo – también conocida como la percepción somatosensorial – y orientación.
- **Lóbulo occipital:** Participa activamente en la percepción y reconocimiento visual. Detecta patrones y procesa la información para enviarla a otras áreas del encéfalo.
- **Lóbulo temporal:** Actúa en el reconocimiento de rostros, está altamente relacionada con la articulación del lenguaje, comprensión de sonidos, voces y música. Actúa también como soporte en la regulación de emociones como la motivación, placer y ansiedad.

- **Lóbulo insular:** Es parte importante del sentido del gusto, control visceral. También participa en la somatopercepción.
- **Lóbulo límbico:** Actúa en la expresión de afectos y la memoria.

Es claro entonces, que el cerebro posee ciertas divisiones funcionales que se encargan en mayor medida de alguna tarea. Éstas son descritas por Gray (2002) en su obra donde reconoce como divisiones funcionales al área sensorial primaria, el área motora primaria y las áreas de asociación.

El área sensorial primaria incluye “el área visual del lóbulo occipital, el área auditiva primaria en el lóbulo temporal y la corteza insular, y el área somatosensorial en el lóbulo parietal” (Gray, 2002). Esta área recibe señales de los nervios sensoriales y las envía a través del tálamo. El área de asociación la define Gray (2002) como la receptora de información entrante de áreas sensoriales y está implicada en el proceso de percepción, pensamiento y toma de decisiones.

Finalmente, se encuentra el área motora primaria, la cual, según Gray (2002), ocupa la parte posterior del lóbulo frontal, delante del área somatosensorial. Esta área envía los axones hasta las neuronas del tronco encefálico y la médula espinal para realizar sus tareas. Sin embargo, hablar de axones y neuronas implica entrar en el terreno de la microarquitectura cerebral, pero de esta forma será posible entender qué “señales” envía el cerebro para la intercomunicación corporal e incluso entre el mismo órgano. Comprender este aspecto será de mayor utilidad en capítulos posteriores de la presente investigación.

7.1.3. El cerebro a nivel microestructura

Es momento de hablar sobre la neurona; componente del sistema nervioso a un nivel de escala menor al de los anteriores constituyentes del sistema nervioso. Las neuronas son parte importante del sistema nervioso, pues incluso “se pueden distinguir dos grupos celulares básicos: células propias del SNC y células comunes con otros sistemas del organismo, como las células endoteliales de los capilares sanguíneos. Las células propias del SNC son de dos tipos diferentes: a) células excitables, denominadas neuronas y b) células no excitables, que incluyen la neuroglia y las células ependimarias” (Ojeda Sahagún & Icardo de la Escalera, 2004).

7.1.3.1. Las neuronas

Las neuronas son un componente principal del sistema nervioso, cuya función principal es recibir, procesar y transmitir información a través de señales químicas y eléctricas gracias a la excitabilidad eléctrica de su membrana plasmática. Las neuronas “son las unidades estructurales y funcionales del sistema nervioso. Son células excitables especializadas en la recepción, integración, transformación y transmisión en una sola dirección (conducción

ortodrómica) de información codificada por cambios electroquímicos en su membrana plasmática” (Ojeda Sahagún & Icardo de la Escalera, 2004).

Estas células “están especializadas en la recepción de estímulos y conducción del impulso nervioso (en forma de potencial de acción) entre ellas mediante conexiones llamadas sinapsis, o con otros tipos de células como, por ejemplo, las fibras musculares de la placa motora. Altamente diferenciadas, la mayoría de las neuronas no se dividen una vez alcanzada su madurez; no obstante, una minoría sí lo hace” (Cayre, Malaterre, Scotto-Lomassese, Strambi, & Strambi, 2010).

Se han realizado estimaciones que señalan que, “el encéfalo humano contiene 86,000,000,000 (ochenta y seis mil millones. 8.6×10^{10}) de neuronas, de las cuales cerca de 10,000,000,000 (diez mil millones. 1×10^{10}) son células piramidales corticales. Estas células transmiten las señales a través de mil billones (10^{15}) de conexiones sinápticas” (Murre & Sturdy, 1995).

7.1.3.2. Morfología de las neuronas

Respecto a la morfología de las neuronas, diferentes autores hacen señalizaciones distintas a la composición de estas células. Ojeda Sahagún e Icardo de la Escalera (2004) mencionan que las neuronas “se caracterizan por poseer una gran superficie celular, lo que les permite llevar a cabo sus funciones específicas. El aumento de la superficie celular se consigue mediante la presencia de expansiones ramificadas que parten del cuerpo celular o soma neuronal y que se denominan neuritas (axón y dendritas). Ciertos tipos de neuronas se caracterizan por la presencia en su citoplasma de pigmentos de diferentes tipos, por lo que sus agrupaciones presentan una coloración característica”. Ellos también destacan como los elementos morfológicos más importantes al soma neuronal, las dendritas y al axón o también denominado cilindroeje.

Aunque otros autores destacan de la morfología de las neuronas “un núcleo voluminoso central, situado en el soma; un pericarion que alberga los orgánulos celulares típicos de cualquier célula eucariota; y neuritas (esto es, generalmente un axón y varias dendritas) que emergen del pericarion” (Paniagua, y otros, 2002).

Incluyendo la morfología que propone Merck & Co Inc. (2022), en la cual propone como componentes principales el cuerpo celular o también llamado soma neuronal, las dendritas y axones, así como las vainas de mielina de las neuronas. Estos componentes pueden observarse en la figura 7.1.4 con una observación gráfica de la morfología de las neuronas.

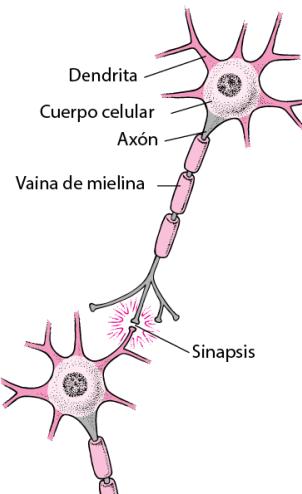


Figura 7.1.4. Morfología de una neurona. (Merck & Co, 2022).

A destacar entre estos componentes morfológicos de la neurona, el primero será el soma neuronal. Esta estructura está localizada en cada una de las neuronas del cerebro, la médula espinal y también en los ganglios espinales del sistema nervioso periférico. La función del soma neuronal es “mantener al núcleo junto con la información genética (ADN), contener a las proteínas del retículo endoplasmático y además es el que genera la energía necesaria para que la neurona trabaje correctamente” (Fisió, 2022). Gracias a esta definición, dado que otros autores definen al cuerpo celular como el encargado de estas funciones, esta parte de la célula recibe ambos nombres, tanto soma neuronal como cuerpo celular.

Es, de hecho, en el soma neuronal donde se asienta el núcleo celular. Tal como lo señalan Ojeda Sahagún e Icardo de la Escalera (2004), pues este elemento “constituye el centro para el crecimiento y mantenimiento de las neuritas y el lugar de la síntesis proteica. El tamaño del soma y del núcleo neuronal se encuentra en relación directa con el número y longitud de neuritas”.

Para comprender la estructura anterior hay que entender qué es una neurita. Kevin Flynn lo define de la manera más concisa, pero adecuada para los términos necesarios en esta investigación; señalando que éstas son “cualquier proyección del soma de una neurona, ya sea una dendrita o un axón. El término se usa con frecuencia al hablar de células nerviosas inmaduras o en desarrollo” (Flynn, 2013). Es por ello que bajo este término, Ojeda Sahagún e Icardo de la Escalera engloban a los axones y las dendritas.

Las neuronas son muy similares a células de otros tejidos en el aspecto de su estructura y organización de su núcleo celular encontrado en el cuerpo de la neurona, pero incluso así, Ojeda Sahagún e Icardo de la Escalera (2004) clasifican en tres tipos de estructuras características al soma neuronal, las cuales son las siguientes:

- Cuerpos de Nissl.
- Neurofilamentos.
- Neurotúbulos.

Otro elemento importante de las neuronas son las dendritas. Las dendritas son “ramificaciones que proceden del soma neuronal que consiste en proyecciones citoplasmáticas envueltas por una membrana plasmática sin envoltura de mielina. En ocasiones, poseen un contorno irregular, desarrollando espinas. Sus orgánulos y componentes característicos son: muchos microtúbulos y pocos neurofilamentos, ambos dispuestos en haces paralelos; además muchas mitocondrias; grumos de Nissl, más abundantes en la zona adyacente al soma; retículo endoplasmático liso, especialmente en forma de vesículas relacionadas con la sinapsis” (Wikipedia, Neurona, 2022).

Asimismo, éstas también se definen como “prolongaciones celulares que se originan a partir del soma neuronal. Constituyen la porción receptiva de las neuronas y suelen ser muy numerosas. Las dendritas que se originan directamente del soma neuronal se denominan dendritas primarias o principales. Éstas se ramifican repetidamente, por lo general de forma dicotómica, dando origen a las dendritas secundarias. El conjunto de dendritas primarias y sus ramificaciones se conoce como árbol dendrítico” (Ojeda Sahagún & Icardo de la Escalera, 2004).

Estas prolongaciones de las neuronas están “dedicadas principalmente a la recepción de estímulos y, secundariamente, a la alimentación celular” (Roche, 2003). Éstas tienen quimiorreceptores que son capaces de reaccionar con los neurotransmisores que son enviados desde los extremos de los axones y siendo altamente importantes para la correcta transmisión de los impulsos quimioeléctricos a través de la vía nerviosa.

Las dendritas contienen la mayor parte de organelos que están presentes en el soma neuronal, entre ellos el retículo endoplásmico; otra característica es que son más cortas, gruesas y rugosas que el axón. “Las rugosidades se deben a la presencia de numerosas expansiones cortas, las espinas dendríticas que constituyen puntos donde una neurona entra en contacto con otras. Las dendritas no poseen envolturas especializadas, como ocurre en el caso de los axones” (Ojeda Sahagún & Icardo de la Escalera, 2004).

Otra parte importante de la neurona es el axón. El axón es definido en algunas ocasiones como “una delgada y extensa prolongación del soma neuronal, que está rodeado por su membrana, el axolema. El axolema puede estar recubierto por células de Schwann en el sistema nervioso periférico de vertebrados, con producción o no de mielina. Puede dividirse, de forma centrífuga al pericarion, en tres sectores: el cono axónico, el segmento inicial y el

resto del axón” (Paniagua, y otros, 2002). Es, de hecho, destacado por Ojeda Sahagún e Icardo de la Escalera (2004) como la prolongación más importante del soma neuronal.

Algunas características notables de los axones, descritas en la obra de Ojeda Sahagún e Icardo de la Escalera (2004), son las siguientes:

- I. El axón carece de la maquinaria precisa para realizar la síntesis proteica, de modo que los neurotransmisores y otros materiales deben ser transportados continuamente desde el soma neuronal a las terminaciones axonales, lo que constituye el transporte o flujo axonal anterógrado; también existe un movimiento en sentido contrario de menor intensidad llamado flujo axonal retrógrado.
- II. A excepción del segmento inicial, el cual es la zona de origen del soma neuronal, la superficie del axón se encuentra cubierta por una vaina de mielina o por una envoltura celular, recibiendo el nombre de axones mielinizados o amielínicos respectivamente. La envoltura celular (para el caso del sistema nervioso central) es de oligodendroglia.

Cabe resaltar que la vaina de mielina la cual rodea al axón “no es continua, sino que se encuentra interrumpida de manera periódica. En estas zonas desnudas, denominadas nódulos de Ranvier, el axón puede poseer ramas colaterales. Los axones suelen terminar ramificándose en varias prolongaciones, las terminaciones axonales o telodendrias, que finalmente establecen contactos o sinapsis con otras neuronas o con células efectoras (músculo o glándulas)” (Ojeda Sahagún & Icardo de la Escalera, 2004).

Este mismo conjunto de autores, junto con otros especialistas en el área, describen también los tres sectores anteriormente mencionados:

- **Cono axónico.** “Adyacente al pericarion, es muy visible en las neuronas de gran tamaño. En él se observa la progresiva desaparición de los grumos de Nissl y la abundancia de microtúbulos y neurofilamentos que, en esta zona, se organizan en haces paralelos que se proyectarán a lo largo del axón” (Paniagua, y otros, 2002).
- **Segmento inicial del axón.** “En él comienza la mielinización externa. En el citoplasma, a esa altura se detecta una zona rica en material electrodenso en continuidad con la membrana plasmática, constituido por material filamentoso y partículas densas. La membrana se continúa con el axolema y se asume que este sector interviene en la generación del potencial de acción que transmitirá la señal sináptica” (Kole & Stuart, 2012).
- **Resto del axón.** Según Paniagua y los autores que apoyaron su obra (2002), es en este sector donde aparecen los nódulos de Ranvier y las sinapsis.

Con base en los autores destacados previamente es posible obtener una definición propia útil para el objetivo de la presente investigación, pues la importancia de estas células en la presente, radica en su composición y comunicación. Para este caso, cabe resaltar que una neurona está compuesta del soma neuronal, las dendritas y los axones, principales componentes de estas células, donde todos estos son importantes para la comunicación entre estas células a través de actividades como la transmisión de potenciales eléctricos y síntesis proteica.

7.1.3.3. Funciones de las neuronas

Sin embargo, hablar de la comunicación entre neuronas requiere también profundizar en las funciones que las neuronas cumplen. No sólo es necesaria la comprensión de estos tópicos, sino que, aunque podría parecer de menor importancia mencionar a algunos investigadores en esta área, para esta investigación tendrán alta relevancia, pues no sólo estarán involucrados en la investigación de estos procesos neuronales, también lo están al hablar de modelos matemáticos de redes que formaron para explicar el complejo sistema de neuronas humano; pero previo a los temas de modelado, será primordial entender las funciones neuronales, explicadas a continuación.

La principal función de las neuronas es comunicarse entre sí. Estas células poseen una capacidad de realizar esta tarea con precisión, rapidez y a larga distancia, sin importar si éstas son de carácter nervioso, muscular o glandular. Es a través de estas células por donde se transmiten señales eléctricas a las cuales se les denomina impulsos nerviosos, los cuales “viajan por toda la neurona comenzando por las dendritas hasta llegar a los botones terminales, que se pueden conectar con otra neurona, fibras musculares o glándulas” (Wikipedia, Neurona, 2022). Esta conexión entre una neurona y otra es lo que recibe el nombre de sinapsis.

Es de hecho la sinapsis la unidad más simple de funcionamiento nervioso segmentario, la cual requiere dos neuronas: una neurona sensitiva receptora y una neurona motora o efectora; así lo explican Afifi y Bergman (1998), quienes también mencionan que “el acoplamiento anatomofuncional de estas dos neuronas ocurre a través de lo que se llama sinapsis. Las arborizaciones terminales de la neurona sensitiva (axones) se dilatan en pequeños brotes o botones (llamados *boutons terminaux*, un término acuñado por un investigador francés), que yacen en contacto con las dendritas, cuerpos celulares o axones de las neuronas efectoras” (Afifi & Bergman, 1998).

Las neuronas “conforman e interconectan los tres componentes del sistema nervioso: sensitivo, motor e integrador o mixto; de esta manera, un estímulo que es captado en alguna región sensorial entrega cierta información que es conducida a través de las neuronas y es

analizada por el componente integrador, el cual puede elaborar una respuesta, cuya señal es conducida a través de las neuronas. Dicha respuesta es ejecutada mediante una acción motora, como la contracción muscular o secreción glandular” (Wikipedia, Neurona, 2022).

La primera función destacable de las neuronas es el impulso nervioso, en el cual las neuronas transmiten ondas eléctricas causadas por un cambio transitorio en la membrana plasmática. La propagación de estas ondas se debe a “la existencia de una diferencia de potencial o potencial de membrana entre la parte interna y externa de la célula, que surge gracias a las concentraciones distintas de iones a ambos lados de la membrana, según describe el potencial de Nernst” (Cromer, 1996). Usualmente esta diferencia de potencial, según Cromer (1996), es de -70 milivoltos (mV).

La carga de una célula inactiva se mantiene en valores negativos, pues se trata de la carga del interior respecto al exterior, y varía de unos estrechos márgenes. “Cuando el potencial de membrana de una célula excitabla se despolariza más allá de un cierto umbral (de 65 mV a 55 mV aproximadamente) la célula genera un potencial de acción. Un potencial de acción es un cambio muy rápido en la polaridad de la membrana de negativo a positivo y vuelta a negativo, en un ciclo que dura milisegundos” (Bear, Connors, & Paradiso, 2002).

El proceso del potencial de acción puede verse ilustrado en la figura 7.1.5, que muestra el voltaje en función del tiempo transcurrido, observando el cambio que ocurre cuando esta función es realizada.

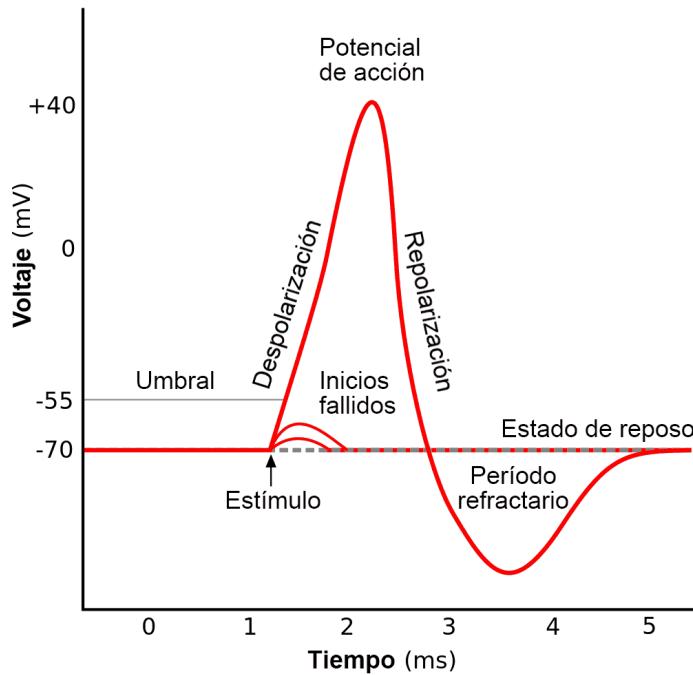


Figura 7.1.5. Gráfica del potencial de acción de una neurona en función del tiempo. (MDurance, 2021).

La siguiente función de las neuronas será en la que se hará mayor énfasis en este documento, pues se trata de las corrientes iónicas que ocurren durante el potencial de acción, siendo de alta importancia para la problemática actual, también se debe a que el primer registro de un potencial de acción fue realizado por los investigadores Alan Lloyd Hodgkin y Andrew Fielding Huxley.

Estos nombres no sólo son notables por el estudio de esta función de las neuronas, sino que también realizaron un conjunto de ecuaciones diferenciales ordinarias no lineales las cuales aproximan las características eléctricas de las neuronas (en general, de las células excitables) proponiendo el modelo en 1952 y posteriormente obteniendo el Premio Nobel en fisiología – medicina por esta investigación en 1963 tras investigar y registrar esta exploración experimentando con el axón de un calamar gigante (Hodgkin & Huxley, 1939).

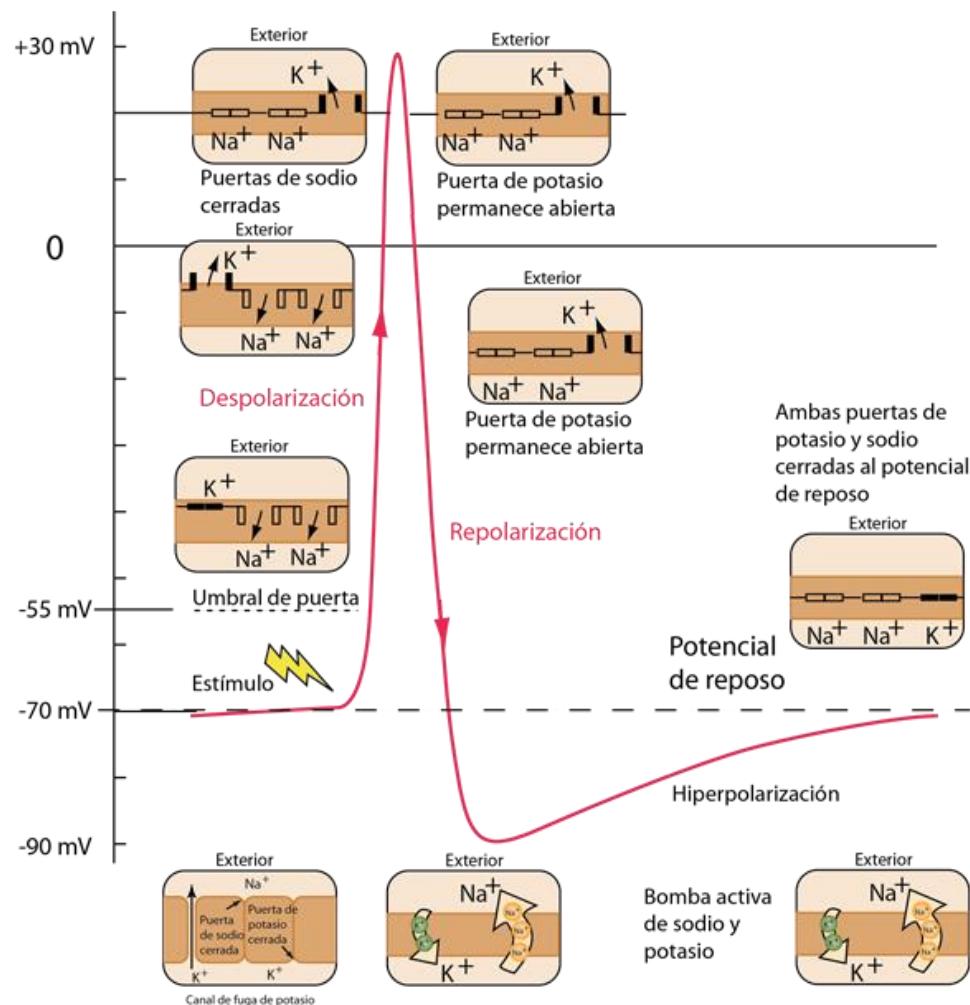


Figura 7.1.6. Gráfica del potencial de acción de una neurona mostrando la interacción iónica. (Olmo, Nave, & Nave, 2022)

El potencial de acción comprende tres fases:

- 1)Potencial en reposo o potencial de membrana, llevándose a cabo una permeabilidad al sodio y al potasio.
- 2)Despolarización de la membrana celular al sodio y al potasio.
- 3)Repolarización de la membrana al sodio y al potasio.

Hodgkin y Huxley midieron la conductancia durante la despolarización y repolarización observando cambios de conductancia para el sodio (Na) y el potasio (K) durante el potencial de acción (Hodgkin & Huxley, 1939).

Al observar los cambios previos puede determinarse la diferencia de potencial por la diferencia absoluta entre las cargas positivas y negativas entre el interior y el exterior con relación a la membrana. Sin embargo, como a continuación se tratarán las cargas eléctricas de las moléculas que atraviesan la membrana, para entender estos conceptos es importante destacar que estas cargas se originan gracias a los iones; átomos o moléculas “cargadas eléctricamente de modo que ya no son eléctricamente neutras” (Ebbing & Gammon, 2010). De este modo, puede haber dos tipos de iones los cuales también describen Ebbing y Gammon, que son los aniones y los cationes. Un anión es un ion con una ganancia de electrones o como se conoce, un ion con carga negativa, así entonces, un catión es un ion con pérdida de electrones, es decir, un ion con carga positiva.

Lo anterior es necesario explicarlo y entenderlo porque de hecho, Wikipedia (2022) en su sitio especializado para la neurona, explica que la diferencia durante el potencial de acción se obtiene por la carga aniónica y catiónica – es decir, por la carga negativa y positiva – de los iones involucrados. A destacar, el anión participante en este proceso es el cloro (Cl^-); por otra parte, los cationes de mayor reconocimiento son el potasio (K^+), magnesio (Mg^{2+}), calcio (Ca^{2+}) y sodio (Na^+). Este procedimiento puede observarse en la figura 7.1.6.

Ahora, cuando un canal iónico es abierto, “el tránsito iónico es a favor de su gradiente electroquímico, esto es, pretende equilibrar el número de iones, independientemente del potencial trasmembrana actual. Este mecanismo circunstancial de movimiento iónico permite el tránsito entre estados de polarización y despolarización” (Wikipedia, Neurona, 2022).

Esto anterior con el objetivo de equilibrar la diferencia electroquímica que existe tanto al exterior como al interior de la membrana, siguiendo, claro, la definición de gradiente en matemáticas, pues se trata de “la variación del valor de una magnitud en dos puntos y la distancia que los separa” (Sáenz, 2013), ya que la definición usada de gradiente electroquímico previamente combina tanto este concepto como su aplicación dentro de la electroquímica, siendo ésta la diferencia iónica en los dos puntos de la membrana anteriormente mencionados.

Este procedimiento fue usado para modelar la propuesta de Hodgkin y Huxley, la cual se retomará más adelante desde un punto de vista más analítico. Finalmente, Wikipedia (2022) menciona otro grupo de funciones importantes de las neuronas, a saber:

- Propiedades electrofisiológicas intrínsecas.
- Neurosecreción.
- Neurodegeneración.

Las anteriores sólo se mencionan de manera superficial dado que no contribuyen en gran medida para solucionar el problema de la presente investigación.

Ha sido un largo camino de neuroanatomía que ha habido que recorrer, sin embargo, es necesario revisar dos tópicos más antes de avanzar al siguiente capítulo y acercarse un poco más al problema que se desea abordar en esta investigación. Estos dos temas son la interacción entre neuronas y, por consiguiente, también las conexiones sinápticas.

Ya son del conocimiento del lector conceptos relacionados al cerebro, como su división a macroescala, así como su funcionamiento a microescala. De esta forma se ha revisado el componente principal de este capítulo: la neurona; observando su morfología y funciones, pero también es importante integrar la neurona con más de este tipo de células y explicar la interacción entre estas células.

7.1.3.4. Interacción entre neuronas

Para comenzar con la interacción entre neuronas, es necesario comprender que “el sistema nervioso procesa la información siguiendo un circuito más o menos estándar. La señal se inicia cuando una neurona sensorial recibe un estímulo externo. Su axón se denomina fibra aferente. Esta neurona sensorial transmite una señal a otra aledaña, de modo que acceda un centro de integración del sistema nervioso. Las interneuronas, situadas en dicho sistema, transportan la señal a través de sinapsis. Finalmente, si debe existir respuesta, se excitan neuronas eferentes que controlan músculos, glándulas u otras estructuras anatómicas. Las neuronas aferentes y eferentes, junto con las interneuronas, constituyen el circuito neuronal” (Randall, Burggren, & French, 1998).

La interacción entre neuronas se da gracias a toda la red de neuronas que están interconectadas. Es así que Randall, Burggren y French (1998) realizan definiciones de estos conceptos, como el mencionado circuito neuronal, el cual se trata de “un conjunto de conexiones sinápticas ordenadas que se produce como resultado de la unión de las neuronas a otras en sus regiones correspondientes tras la migración neuronal”.

Mencionando también que “el crecimiento dirigido de los axones y el reconocimiento de las estructuras sinápticas está mediado por el cono de crecimiento, que es una especialización en el extremo de cada axón en crecimiento. El cono de crecimiento detecta y responde a moléculas de señalización que pueden ser de retraimiento, giro o continuación, que identifican las vías correctas, prohíben las incorrectas y facilitan la formación de sinapsis” (Randall, Burggren, & French, 1998).

Cuando se habla de una red neuronal biológica se destaca que “la función de un determinado grupo de neuronas es alcanzar un determinado estado final en función de los estímulos externos. Por ejemplo, en la percepción del color, un grupo de neuronas puede encargarse de acabar en un determinado estado si el estímulo es “rojo” y otro determinado estado si el estímulo es “verde”. El número de “estados estables” posibles del circuito neuronal se corresponde con el número de patrones (en este caso colores diferentes) que puede reconocer el circuito neuronal. Los trabajos de Freeman en los años 1990 aclararon que un determinado grupo de neuronas sigue un patrón de evolución temporal caótico hasta alcanzar un determinado estado” (Solé & Manrubia, 1996).

Solé y Manrubia (1996) profundizan mucho en estos tópicos de circuitos neuronales. De hecho también entran en análisis de modelos matemáticos, pues por ejemplo, estudian algunos temas de estas redes como la teoría de Hopfield y la regla de Hebb pues de estos dos argumentos mencionan que “estiman la relación entre el número de neuronas N que intervienen en reconocer p patrones y la probabilidad de error P_e en el reconocimiento de patrones”.

Sin embargo, la teoría de Hopfield también se trata de un modelo matemático y ciertamente en este capítulo ya se están tocando temas que corresponden a apartados posteriores. Aun así, los conocimientos de neuroanatomía han sido cubiertos en su totalidad y todos y cada uno de los conceptos vistos en este capítulo serán de alta importancia en los posteriores.

Aunque ya se están tocando temas de matemáticas, antes es necesario conocer cómo es posible obtener la actividad eléctrica cerebral, de qué forma se puede clasificar la actividad una vez que se capture y cuáles serían los aparatos con los que se obtendrían estos datos. Estos conceptos y más relacionados serán del conocimiento del lector en el próximo capítulo relacionado a la electroencefalografía.

7.2. Electroencefalografía y estudios del cerebro

Todo hombre puede ser, si se lo propone, escultor de su propio cerebro
Santiago Ramón y Cajal

Antes de avanzar con algunos conocimientos computacionales, mantendremos la línea teórica centrada en el cerebro, pero hablando de un dispositivo importante para el desarrollo de esta investigación el cual sustenta la comprensión de un área conocida como la electroencefalografía.

7.2.1. El electroencefalograma

La herramienta principal de la electroencefalografía es el electroencefalograma. Éste puede definirse como la herramienta que permite realizar una exploración neurofisiológica basándose en el registro de la actividad bioeléctrica cerebral. Propiamente puede definirse como “un estudio de la función cerebral que recoge la actividad eléctrica del cerebro en situación basal y con métodos de activación, como la hiperventilación y la fotoestimulación” (Urrestarazu, 2022). Este estudio recopila la señal eléctrica la cual posteriormente se amplifica en forma de líneas, interpretándose la actividad de las distintas áreas cerebrales a lo largo del tiempo.

Este estudio es definido también como “una técnica neurofisiológica para detectar y analizar la actividad eléctrica en el cerebro. El procedimiento consiste en colocar varios electrodos en el cuero cabelludo” (Quiroga Subirana, 2013).



Figura 7.2.1. Ejemplo de electroencefalograma (Sosa Romano, 2022).

Quiroga Subirana (2013) menciona que esta técnica consiste en “colocar alrededor de 20 electrodos al cuero cabelludo. Cada electrodo envía una señal a una máquina llamada electroencefalógrafo, que muestra la fluctuación rítmica de la actividad eléctrica del cerebro (ondas cerebrales) visualmente como una línea ondulante. De esta manera, es posible que se controle la actividad cerebral” (Quiroga Subirana, 2013). Un ejemplo de éste se aprecia en la figura 7.2.1.

El estudio con un electroencefalograma, o desde ahora llamado EEG, permite observar patrones normales o anormales del comportamiento cerebral que pueden hacer sospechar lesiones o enfermedades características. Un EEG sirve para “para observar el funcionamiento eléctrico cerebral. Es de interés conocer su normalidad o no en pacientes con alteración de las funciones cerebrales, bien de forma persistente o bien de modo episódico” (Urrestarazu, 2022).

Debido a esto, el estudio con un EEG es “un medio de diagnóstico funcional de enfermedades cerebrales complementario a otros estudios, especialmente los radiológicos” (Urrestarazu, 2022) como es el caso de la resonancia magnética.

Un EEG puede detectar alteraciones de todo el cerebro o de algunas zonas. Esto es útil para revisar alteraciones en lesiones, como tumores, hemorragias, encefalitis, traumatismos u otro tipo, así como lesiones difusas tóxicas, metabólicas o infecciosas.

Este radar mental permite detectar la actividad cerebral de las neuronas de los dos hemisferios cerebrales y de los lóbulos del cerebro, tal como lo menciona Iranzo de Riquer (2022) quien además añade que con un EEG “no puede detectar la actividad de las neuronas de las áreas profundas del cerebro, ni del cerebelo ni del tronco del encéfalo. Las neuronas cerebrales se comunican a través de impulsos eléctricos y están siempre activas, tanto si la persona está despierta como dormida. Esta actividad eléctrica se manifiesta en forma de ondas de distinta intensidad: baja, moderada y alta” (Iranzo de Riquer, 2022).

Una prueba EEG, según Urrestarazu (2022), normalmente arroja las siguientes indicaciones con mayor frecuencia:

- Cefaleas.
- Epilepsia.
- Trastornos del movimiento
- Trastornos del sueño

Paniagua Soto (2016) también coincide con las indicaciones que un EEG puede regresar, así como añade que el estudio especialmente en niños y recién nacidos es fundamental para darle seguimiento a la maduración cerebral o a posibles epilepsias infantiles, pues, en definitiva señala esta herramienta como ciertamente útil.

7.2.2. Breve historia del electroencefalograma

Wikipedia (s.f.) en su sitio “Electroencefalografía” plasman una breve historia del electroencefalograma. Ellos denotan como primer hecho importante los hallazgos presentados por Richard Birmick Caton sobre los fenómenos bioeléctricos en los hemisferios cerebrales de ratones y monos en 1875.

El siguiente hecho rescatable en el sitio previamente mencionado data de 1912 donde Vladimir Pravdich-Neminsky publica el primer EEG y potenciales evocados de perros. Posteriormente Hans Berger comenzó a realizar sus estudios sobre electroencefalografía en humanos en 1920.

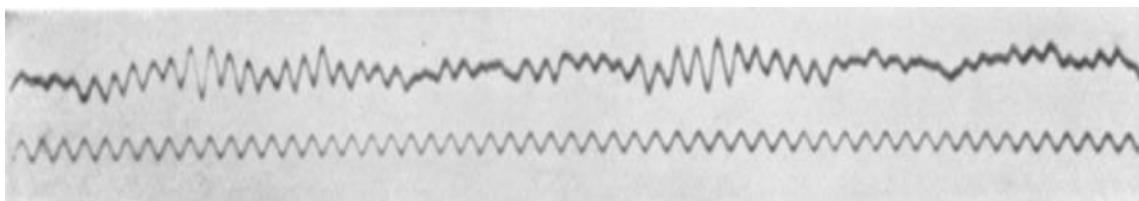


Figura 7.2.2. Primer registro de un electroencefalograma en un humano (Wikipedia, s.f.).

En la figura 7.2.2 se observa el primer electroencefalograma de un humano registrado, perteneciente al año 1929. En este mismo año, Hans Berger acuñó el término “electroencefalograma” para “describir el registro de las fluctuaciones eléctricas en el cerebro captadas por unos electrodos fijados al cuero cabelludo” (Ramos-Argüelles, 2009).

7.2.3. Ondas encefálicas

Las ondas cerebrales o también llamadas ondas encefálicas son, según Wikipedia (2022) en su sitio web “Ondas cerebrales”, registros de “actividad eléctrica que generan las diferentes estructuras del encéfalo. Estas oscilaciones son de muy baja amplitud, del orden de microvoltios en humanos y no siempre siguen una sinusoides regular. El análisis de los patrones de ondas, es una gran herramienta de investigación en la Neurofisiología del estado de funcionamiento del encéfalo”.

Estas oscilaciones se originan en distintas estructuras tales como el cerebro, cerebelo y núcleos basales. Lo que permite un EEG es recopilar la suma de potenciales de las dendritas postsinápticas. Éstas “forman un agrupamiento de unidades (grupo neuronal o región cortical) con orientación similar y proximidad inmediata, en las capas superficiales de la corteza” (Barrett, Barman, Boitano, & Brooks, 2013).

Según Hermann (1997), las ondas de la actividad eléctrica de fondo del encéfalo que se detectan se pueden clasificar según su frecuencia de oscilación de la forma en la que se presenta a continuación en la tabla 7.2.1.

Tabla 7.2.1. Clasificación de las ondas encefálicas (Hermann, 1997).

Ondas gamma	25 a 100 Hz
Ondas beta	14 a 30 Hz
Ondas alfa	8 a 13 Hz
Ondas theta	3 a 7 Hz
Ondas delta	0 a 4 Hz

Por último, para finalizar esta breve sección teórica cabe destacar que existen múltiples métodos de activación encefálica entre los cuales menciona Wikipedia (s.f.):

- Hiperpnea
- Estimulación luminosa intermitente
- Estimulación visual
- Estimulación auditiva
- Estimulación somestésica
- Estimulación nociceptiva

7.2.4. Interfaces cerebro – computadora

Regresaremos un paso atrás en la evolución de estas interfaces antes de hablar de las interfaces cerebro – computadora, pues, anterior a estas interfaces es fundamental mencionar el medio de comunicación entre humanos y computadoras del que se deriva esta tendencia más novedosa y es que se trata de la interfaz humano – computadora.

7.2.4.1. El ancestro: interfaces humano – computadora

Esta interacción, también conocida como IHC o como se le denominará de aquí en adelante, HCI de sus siglas en inglés “*human – computer interface*” estudia la comunicación entre humanos y computadoras y gira en torno a la idea de “hacer que el envío de información entre humano - computador sea más eficiente, y una de sus metas más importantes es hacer más eficiente el trabajo que los humanos realizan con las computadoras” (Reyes Núñez, Soto Gómez, & Vicario Solórzano, 2022).

Dentro de esta disciplina se consideran aspectos importantes como los que se mencionan en el sitio web de Wikipedia especializado a la interacción persona – computadora, donde mencionan “el diseño, evaluación e implementación de los aparatos tecnológicos interactivos, estudiando el mayor número de casos que les pueda llegar a afectar. El objetivo es minimizar errores, incrementar la satisfacción, disminuir la frustración y, en definitiva, hacer más

productivas las tareas que rodean a las personas y las computadoras" (Wikipedia, Interacción persona-computadora, s.f.).

Reyes Núñez, Soto Gómez y Vicario Solórzano (2022) mencionan que los componentes existentes en una HCI son los usuarios, las computadoras y la interacción que existe entre ellos; en otros medios como el sitio de Wikipedia, la información al respecto respalda a estos autores, con la ligera diferencia en el último componente mencionado, que se le da el nombre de proceso interactivo, sin embargo, su idea representativa es la misma.

El área de HCI es muy extensa; en ella podemos encontrar conocimiento que abarca principios y metodologías de diseño, disciplinas e incluso la investigación que se está llevando a cabo en este campo. Estos tópicos son fundamentales para entender el estudio de la comunicación entre personas y computadoras que, para efectos de esta investigación, se mencionarán brevemente a continuación pues el propósito es usarlas como apoyo para las interfaces cerebro – computadora, que es el medio de interés de esta exploración.

Una mención breve a los principios de diseño en el área de HCI es la que menciona Green (2008) dando tres principios que, resumiendo su tratamiento, expone lo siguiente:

- Fijar quién será el usuario y su tarea. Se trata de apuntar a un público objetivo y determinar de la misma forma las tareas que llevarán a cabo.
- Medidas empíricas. Este principio trata de establecer ciertas especificaciones cuantitativas como formas de medida de eficiencia de la interacción deseada; el autor da como ejemplos el número de usuarios necesarios para realizar una tarea, el tiempo necesario para completarla o el número de errores producidos durante la realización de la misma.
- Diseño iterativo. La idea del autor se solidifica sobre volver a empezar el proceso para modificar el diseño, probarlo, analizar los resultados y repetir cíclicamente el procedimiento en busca de mejorar esta interfaz de forma continua.

Ahora, como metodologías del diseño de estas interfaces, distintos autores han aportado a este tema, no obstante, en modelos modernos, se ha buscado centrar las miradas en retroalimentación a los usuarios, mejorar la comunicación y con ello mejorar la experiencia deseada.

Lo anterior lo expone Kaptelinin (2012) de la misma forma que el propone una metodología de diseño a la que llama teoría de la actividad, que en su definición, comenta que se usa para "definir el contexto en el que tiene lugar la interacción entre personas y ordenadores. Proporciona un marco de referencia para razonar sobre acciones en estos contextos, herramientas analíticas en forma de listas de tareas que los investigadores deberían tener en

cuenta y toma parte en el diseño de interacción desde una perspectiva centrada en la actividad" (Kaptelinin, 2012).

Esta información la complementa Tidwell (1999) quien añade dos rasgos más a la información dada por Kaptelinin, pues ella expone otra metodología denominada el diseño centrado en el usuario que está teniendo gran apoyo después de acuñarse. Se trata de una metodología donde el usuario es el centro del diseño y el objetivo es articular el sistema a medida del consumidor estudiando las necesidades y limitaciones de los interesados.

Tidwell (1999) también añade dentro de este campo lo que se conoce como los principios de diseño de la interfaz de usuario. Ella declara que deben considerarse siete principios al diseñar una interfaz de usuario, siendo éstos los siguientes:

- Tolerancia.
- Simplicidad.
- Visibilidad.
- Factibilidad.
- Consistencia.
- Estructura.
- Retroacción.

Sin profundizar en los principios de diseño de las interfaces de usuario, ha de concluirse que estas tres metodologías de diseño se tomarán en cuenta como parte importante en el diseño de una HCI.

Por último, para cerrar los aspectos destacables de las HCI cabe mencionar que su extensión está creciendo que el campo continúa especializándose más y más, tal como se menciona en el sitio de Wikipedia (s.f.) especializado en interfaces persona – computadora, este conjunto de conocimientos ya posee ciertas disciplinas enfocadas en aspectos más técnicos. A mencionar algunas encontradas en el previo artículo están:

- Características visuales.
- Características auditivas.
- Entorno de tareas.
- Entorno de máquinas.
- Áreas de interfaz.
- Flujo de entradas.
- Salidas.
- Retroalimentación.

Es así que observamos que el estudio de las interfaces humano – computadora influyen en la comunicación básica entre una persona y una computadora, que al pasar el tiempo, se ha especializado buscando formas de mejorar esta circulación de información y de la que desprenden conceptos primordiales para el diseño y elaboración de sistemas que necesiten este intercambio de información entre personas y computadoras.

Estos conceptos de las HCI son los que preparan el camino para la llegada de las interfaces cerebro – computadora o BCI llamadas a partir de ahora por sus siglas en inglés “*brain – computer interface*”. Una BCI es una tecnología basada en la adquisición de información neuronal, según se menciona en el artículo de Wikipedia (s.f.) de las interfaces cerebro – computadora, donde también mencionan cómo esta tecnología permite instaurar un canal de comunicación entre persona y computadora que puede aprovechar información motora, cognitiva y emocional obtenida directamente del cerebro.

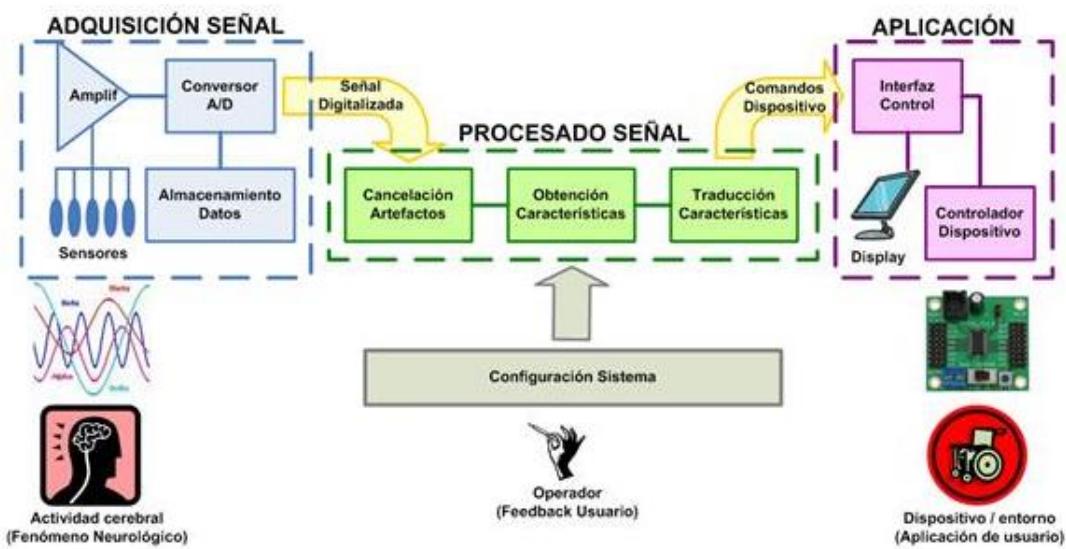


Figura 7.2.3. Modelo funcional genérico de una BCI (Wikipedia, Interfaz cerebro - computadora, s.f.).

En este mismo artículo mencionan un modelo funcional genérico el cual explica el funcionamiento básico de una BCI diferenciando cuatro bloques principales que pueden apreciarse en la figura 7.2.3 e igualmente estos cuatro bloques fundamentales puntualizan a continuación:

- ✓ Adquisición de la señal. En este bloque se obtiene la señal, se amplifica y se realiza una conversión de señal analógica a señal digital. Normalmente los sistemas BCI trabajan a tiempo real, pero opcionalmente también incluye la posibilidad de realizar un registro de esta señal para un estudio posterior de ésta.
- ✓ Procesado de la señal. Aquí se buscan extraer las características de interés de la señal digitalizada en el paso anterior de modo que el dispositivo sea capaz de interpretar las órdenes del usuario. Según Wikipedia (s.f.) se distinguen tres etapas en este bloque:

- Cancelación de artefactos. Se busca eliminar ruido proveniente de otras actividades bioeléctricas que distorsionen la señal.
 - Obtención de características. La señal de entrada es traducida a un vector de características en relación con el fenómeno neurológico que está asociado a la señal.
 - Traducción de características. El vector de características es transformado a una señal de control adecuada para el dispositivo que desea operarse.
- ✓ Aplicación. En este bloque se recibe la señal de control y realiza las acciones correspondientes en el dispositivo.
- ✓ Configuración. Este apartado está hecho para definir y controlar algunos parámetros del sistema.

Las aplicaciones de los sistemas BCI se extienden en muchos campos; en concreto pueden encontrarse dos campos prácticos donde aprovechan altamente este recurso. El primer entorno aplicativo es el ámbito médico, pues estas tecnologías “permiten la rehabilitación de personas con desórdenes neurológicos como la epilepsia, el trastorno por déficit de atención o hiperactividad o la comunicación con personas con parálisis cerebral” (Palucci, y otros, 2023).

Otro entorno de aplicación de sistemas BCI es dentro del ocio y los videojuegos. Existe un ejemplo en este campo que es el “Mindball. Este dispositivo permite el movimiento de una pelota por un tablero y lo hace mediante la detección de relajación del usuario” (Wikipedia, Interfaz cerebro - computadora, s.f.).

En el artículo anterior también se da como ejemplo otro juego que consiste en “el del movimiento de un avatar en un entorno virtual donde las señales cerebrales pueden decidir si el avatar avanza o retrocede, gira a la derecha o a la izquierda, entre otras órdenes básicas como en el caso de Second Life” (Wikipedia, Interfaz cerebro - computadora, s.f.).

Finalmente, en el artículo anterior se menciona que “otro tipo de interfaz, además de interpretar acciones básicas, también se basan en el estado emocional de usuario, siendo capaces de reconocer la excitación, tensión, aburrimiento, meditación, frustración, inmersión como es el caso del dispositivo Emotiv EPOC” (Wikipedia, Interfaz cerebro - computadora, s.f.).

Estos métodos nos ayudarán posteriormente a la estimulación de los sujetos experimentales y tener el conocimiento de estos elementos los cuales serán utilizados como el pilar fundamental de la experimentación de la presente investigación.

Con esto dicho, es momento de pasar al uso de las ciencias de la computación que complementarán altamente la investigación actual, brindando la parte de automatización y predicción al proyecto para la solución del problema de clasificación de intención del movimiento.

7.3. Fundamentos computacionales, inteligencia artificial y aprendizaje automático

Toda tecnología lo suficientemente avanzada es indistinguible de la magia
Arthur C. Clarke

Hemos dejado atrás los conceptos de neurociencia – aunque dejar atrás no significa olvidar – y es momento de revisar lo que, para el autor, son los temas, tanto de mayor interés como los que más le atan en función de obtener el grado profesional como propósito de esta investigación. Se trata de nada más y nada menos que los conceptos matemáticos y computacionales que rodean a esta investigación.

Nos hemos acercado a revisar el cerebro y su estructura, también su estudio a una gran escala y a una pequeña escala y fue aquí donde nos adentramos en nuestro componente más importante: la neurona. Revisamos la morfología de una neurona y las funciones que realiza así como la manera en la que una de estas células interactúa con otra.

Después de todo esto avanzamos de capítulo para entender cómo podía estudiarse esta actividad cerebral y entender lo que era un electroencefalograma, así como la forma en la que se reciben los datos de la actividad cerebral por medio de este aparato; clasificamos las diferentes ondas cerebrales para entender por separado lo que cada tipo de onda representaba. Pero ahora que hablaremos de conceptos de ciencias computacionales, ¿todo esto qué tiene que ver con la computación?

Bien, pues a lo largo del capítulo se observará la importancia de la computadora para poder tratar los datos recibidos por un electroencefalograma, siendo este aspecto la última pieza del presente rompecabezas al cual le he nombrado tesis de grado licenciatura para poder llegar al problema de investigación, el cual tratará justamente de lo que es posible realizar con los datos de un electroencefalograma al manipularlos, entenderlos y aplicar técnicas matemáticas adaptadas a las máquinas para que estos procedimientos los realicen de forma automática, pero para ello, primero será necesario arrancar desde el principio definiendo lo que una computadora es.

7.3.1. Introducción a la computación

Dado que la presente investigación requiere avanzar a tópicos matemáticos y computacionales considerados algo avanzados, estas primeras definiciones no serán muy discutidas, pero son necesarias de mencionar.

7.3.1.1. Computación

Primeramente, hay que definir lo que es una computadora. Una computadora es “un dispositivo electrónico, utilizado para procesar información y obtener resultados; capaz de ejecutar cálculos y tomar decisiones. En el sentido más simple una computadora es un dispositivo para realizar cálculos o computar” (Joyanes Aguilar, 2008).

Indiscutiblemente, para los efectos de esta investigación, la definición de Joyanes Aguilar es completa y concisa para abarcar los términos necesarios de lo que es una computadora, pues se trata de una máquina a la que se le ingresan datos y ésta responde procesándolos y regresando un resultado después de su procesamiento. Joyanes Aguilar (2008) también hace la clarificación de dividir una computadora en dos partes distintas: “hardware y software. El hardware es la computadora en sí misma. El software es el conjunto de programas que indican a la computadora las tareas que debe realizar” donde puede interpretarse esta mención al hardware como la parte tangible de la computadora mientras que el software, siendo los programas del equipo, se vuelve la parte intangible de la computadora. Así también se conoce con el término programador a la persona que escribe programas.

Sin embargo, es justo gracias al software donde en su mayoría puede enfocarse el procesamiento de datos, o más bien, las instrucciones para este procesamiento, pues el procesamiento de los datos que la computadora recibe se realiza gracias a “un conjunto de instrucciones denominadas programas de computadora. Estos programas controlan y dirigen a la computadora para que realice un conjunto de acciones (instrucciones) especificadas por personas especializadas” (Joyanes Aguilar, 2008) a los cuales este autor denomina programas de computadora.

Son justamente estos programas de computadora los que reciben datos, realizan el procesamiento y producen un resultado con los datos recibidos. “Los datos y la información se pueden introducir en la computadora por una entrada (input) y a continuación se procesan para producir una salida (output, resultados)” (Joyanes Aguilar, 2008).

7.3.1.2. Programación

“La computadora se puede considerar como una unidad en la que se colocan ciertos datos (entrada de datos), se procesan y se produce un resultado (datos de salida o información)” (Joyanes Aguilar, 2008). Este autor menciona que los datos, tanto de entrada como de salida pueden ser de cualquier tipo como texto, dibujos, sonido, imágenes, entre muchos otros tipos. Aunque también menciona que el formato más sencillo de comunicación entre una persona con una computadora es mediante un teclado, una pantalla o monitor y un ratón o mouse.

Dado que en esta investigación lo que se producirá es una solución de software, se centrará específicamente en este aspecto de las ciencias computacionales. Para que el hardware realice las operaciones necesarias se especifican por un conjunto de instrucciones llamados programas. “El software de una computadora es un conjunto de instrucciones de programa detalladas que controlan y coordinan los componentes hardware de una computadora y controlan las operaciones de un sistema informático. El auge de las computadoras el siglo

pasado y en el actual siglo XXI, se debe esencialmente al desarrollo de sucesivas generaciones de software potentes y cada vez más amistosas" (Joyanes Aguilar, 2008).

Un programa tiene que escribirse en un lenguaje de programación que será la principal herramienta del programador para indicarle a la computadora qué deberá realizar al escribir el código fuente de un programa. Sin embargo, "la computadora no entiende directamente los lenguajes de programación, sino que se requiere un programa que traduzca el código fuente a otro lenguaje que sí entiende la máquina directamente, pero muy complejo para las personas; este lenguaje se conoce como lenguaje máquina y el código correspondiente código máquina. Los programas que traducen el código fuente escrito en un lenguaje de programación a código máquina se denominan traductores" (Joyanes Aguilar, 2008).

7.3.1.3. El algoritmo

Antes de escribir el código fuente de un programa debe de analizarse el problema y abstraer las partes más importantes que componen a la problemática. Esto es diseñar un algoritmo. Joyanes Aguilar (2008) define a un algoritmo como un método para resolver un problema y él también propone que los pasos para la resolución de un problema son los siguientes:

- 1) Diseño del algoritmo. Describe la secuencia ordenada de pasos —sin ambigüedades— que conducen a la solución de un problema dado. (Análisis del problema y desarrollo del algoritmo).
- 2) Expresar el algoritmo como un programa en un lenguaje de programación adecuado. (Fase de codificación).
- 3) Ejecución y validación del programa por la computadora.

Un algoritmo es independiente del lenguaje de programación y también lo es de la computadora que lo ejecuta y, de hecho, el algoritmo posee una alta importancia en este campo de estudio pues "en la ciencia de la computación y en la programación, los algoritmos son más importantes que los lenguajes de programación o las computadoras. Un lenguaje de programación es tan sólo un medio para expresar un algoritmo y una computadora es sólo un procesador para ejecutarlo. Tanto el lenguaje de programación como la computadora son los medios para obtener un fin: conseguir que el algoritmo se ejecute y se efectúe el proceso correspondiente" (Joyanes Aguilar, 2008).

Todo algoritmo debe cumplir con unas características muy específicas para clasificarse como tal. Estas características necesarias dentro de todo algoritmo, según Joyanes Aguilar (2008) se describen a continuación:

- Un algoritmo debe ser preciso e indicar el orden de realización de cada paso.

- Un algoritmo debe estar bien definido. Si se sigue un algoritmo dos veces, se debe obtener el mismo resultado cada vez.
- Un algoritmo debe ser finito. Si se sigue un algoritmo, se debe terminar en algún momento; o sea, debe tener un número finito de pasos.

7.3.2. Inteligencia artificial

Sin embargo, el desarrollo de software ha evolucionado a lo largo de los años y esto ha traído notables avances en este campo, entre ellos la aparición de la inteligencia artificial o abreviada, IA. Algunos investigadores en el área encuentran que el trabajo en este campo comenzó poco después de la Segunda Guerra Mundial y se le dio su nombre a esta área en 1956. Aun así, la IA ha crecido mucho en los pocos años que ha tenido de desarrollo, con campos que se han creado tales como aprendizaje y percepción, demostración de teoremas matemáticos, diagnóstico de enfermedades, entre muchas otras más. El fundamento de la IA es que “sintetiza y automatiza tareas intelectuales y es, por lo tanto, potencialmente relevante para cualquier ámbito de la actividad intelectual humana. En este sentido, es un campo genuinamente universal” (Russell & Norvig, 2004).

No obstante, la inteligencia artificial no tiene una definición sencilla, pues distintos autores dan definiciones muy dispersas. Russell (2004) realiza un compendio de algunas definiciones en un cuadro que remarca cuatro enfoques que combinan la racionalidad y la forma de actuar de los humanos con el pensamiento y el actuar, lo que resulta en el cuadro presentado en la tabla 7.3.1.

Tabla 7.3.1. Cuadro de definiciones de la inteligencia artificial presentado por Stuart Russell. (Russell & Norvig, 2004).

Sistemas que piensan como humanos	Sistemas que piensan racionalmente
<p>«El nuevo y excitante esfuerzo de hacer que los computadores piensen... máquinas con mentes, en el más amplio sentido literal». (Haugeland, 1985)</p> <p>«[La automatización de] actividades que vinculamos con procesos de pensamiento humano, actividades como la toma de decisiones, resolución de problemas, aprendizaje...» (Bellman, 1978)</p>	<p>«El estudio de las facultades mentales mediante el uso de modelos computacionales». (Charniak y McDermott, 1985)</p> <p>«El estudio de los cálculos que hacen posible percibir, razonar y actuar». (Winston, 1992)</p>
Sistemas que actúan como humanos	Sistemas que actúan racionalmente
<p>«El arte de desarrollar máquinas con capacidad para realizar funciones que cuando son realizadas por personas requieren de inteligencia». (Kurzweil, 1990)</p> <p>«El estudio de cómo lograr que los computadores realicen tareas que, por el momento, los humanos hacen mejor». (Rich y Knight, 1991)</p>	<p>«La Inteligencia Computacional es el estudio del diseño de agentes inteligentes». (Poole <i>et al.</i>, 1998)</p> <p>«IA... está relacionada con conductas inteligentes en artefactos». (Nilsson, 1998)</p>

Afortunadamente, en 1950, Alan Turing apareció en esta ecuación para definir a la inteligencia artificial desde un punto de vista operacional. “En vez de proporcionar una lista larga y quizá controvertida de cualidades necesarias para obtener inteligencia artificialmente, él sugirió una prueba basada en la incapacidad de diferenciar entre entidades inteligentes indiscutibles y seres humanos” (Russell & Norvig, 2004). De este modo, una computadora debería poseer las siguientes capacidades para superar la acuñada Prueba de Turing:

- Procesamiento de lenguaje natural que le permita comunicarse satisfactoriamente en algún idioma.
- Representación del conocimiento para almacenar lo que se conoce o siente.
- Razonamiento automático para utilizar la información almacenada para responder a preguntas y extraer nuevas conclusiones.
- Aprendizaje automático para adaptarse a nuevas circunstancias y para detectar y extrapolar patrones.

Los fundamentos de la inteligencia artificial se han apoyado en muchas otras ciencias, como la filosofía, las matemáticas, la economía, la neurociencia, la psicología, la ingeniería computacional, la teoría del control y cibernetica y la lengua para entender todos estos aspectos que rodean al campo de la IA pero esto no ha detenido el avance de los modelos de inteligencia artificial. Entre ellos, Stuart Russell en su obra del 2004 señala una basta cantidad de modelos que se estudian comúnmente y que él expuso en su libro, entre ellos se encuentran modelos de aprendizaje estadístico, como los modelos de Bayes Simples (o también conocidos como naive Bayes), modelos de aprendizaje con variables ocultas, aprendizaje no supervisado, redes bayesianas con variables ocultas, modelos de Markov, aprendizaje basado en instancias, modelos de vecinos más cercanos, redes neuronales, máquinas núcleo y muchos otros.

7.3.3. Aprendizaje automático

Es tan amplio el campo de la inteligencia artificial que dentro de éste se encuentran ramas notables; la más notable y útil para el caso de la investigación actual es la del aprendizaje automático. Este campo de estudio tiene como objetivo, según Rusell y Norvig (2004), “desarrollar técnicas que permitan que las computadoras aprendan. Se dice que un agente aprende cuando su desempeño mejora con la experiencia y mediante el uso de datos; es decir, cuando la habilidad no estaba presente en su genotipo o rasgos de nacimiento”. Estos mismos autores, en ediciones más recientes de la obra mencionan cómo ocurre el aprendizaje automático, explicando que la computadora observa datos después construye un modelo basado en estos datos y lo utiliza como hipótesis acerca del mundo, siendo ésta, una pieza de software que puede resolver problemas.

Así como la mera inteligencia artificial es tan diversa, el campo del aprendizaje automático también mantiene esta tendencia, pues se ha mantenido en constante desarrollo; tan es así que esta rama tiende a clasificar la investigación y aplicación generada en grupos de modelos de aprendizaje, así como algoritmos de aprendizaje. Flach (2012) en su obra menciona algunos modelos destacables por la forma en la que éstos abordan los problemas.

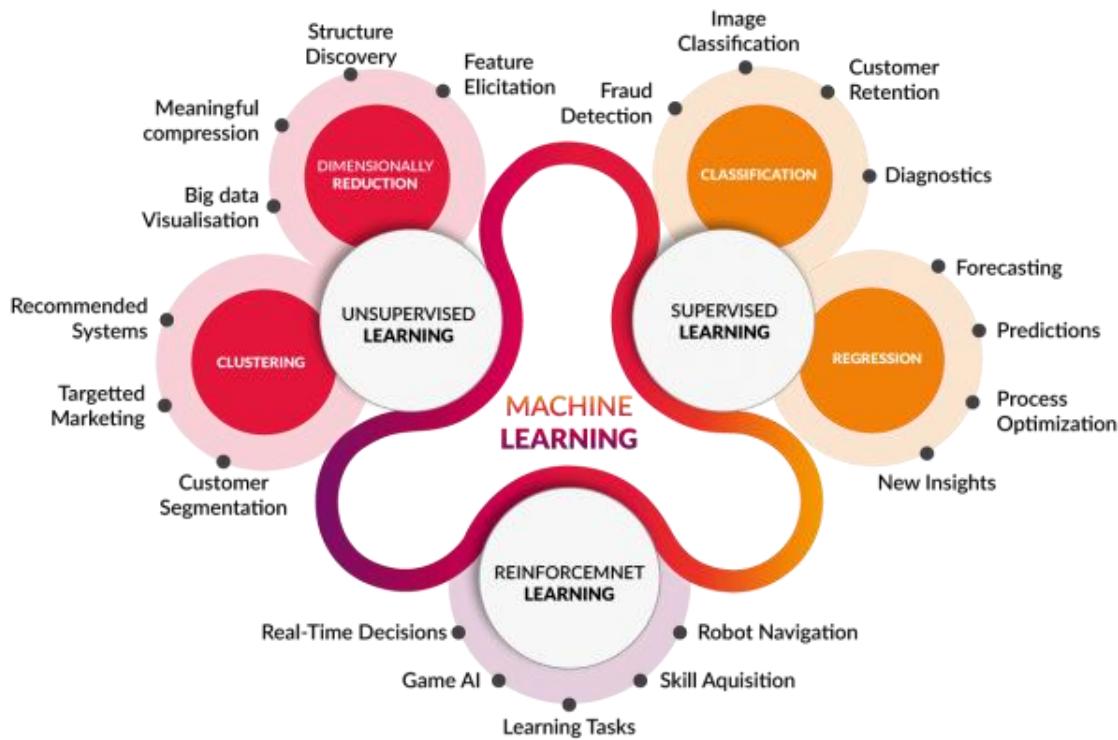


Figura 7.3.1. Clasificación del machine learning (o aprendizaje automático) en sus tres grandes tipos de aprendizaje (González Barrio, Calleja Ochoa, Gómez-Escudero, Rodríguez Ezquerro, & López de Lacalle Marcaide, 2021).

Sin embargo, la riqueza de este campo recae en los distintos tipos de aprendizaje que se han desarrollado bajo el estudio de esta área siendo tres principales:

- 1) **Aprendizaje supervisado:** En este tipo de aprendizaje “el operador proporciona al algoritmo de aprendizaje automático un conjunto de datos conocidos que incluye las entradas y salidas deseadas, y el algoritmo debe encontrar un método para determinar cómo llegar a esas entradas y salidas” (APD, 2019). Esto quiere decir que el algoritmo busca cómo establecer una correspondencia entre las entradas y las salidas deseadas del sistema, donde los ejemplos comunes de estos sistemas son el uso de sistemas con bases de conocimiento las cuales están conformadas por ejemplos anteriormente etiquetados.
- 2) **Aprendizaje no supervisado:** Estos algoritmos están diseñados con el propósito de que el sistema estudie los datos en búsqueda de patrones debido a que “no hay una clave de

respuesta o un operador humano para proporcionar instrucción. En cambio, la máquina determina las correlaciones y las relaciones mediante el análisis de los datos disponibles" (APD, 2019). Entre más datos evalúe el sistema, la toma de decisiones del algoritmo mejorará consiguiendo una mejor segmentación de datos en distintos grupos.

- 3) **Aprendizaje por refuerzo:** El aprendizaje por refuerzo "se centra en los procesos de aprendizajes reglamentados, en los que se proporcionan algoritmos de aprendizaje automáticos con un conjunto de acciones, parámetros y valores finales. Al definir las reglas, el algoritmo de aprendizaje automático intenta explorar diferentes opciones y posibilidades, monitorizando y evaluando cada resultado para determinar cuál es el óptimo" (APD, 2019). Este tipo de sistemas aprenden a través del ensayo y error observando el mundo que les rodea y evaluando la retroalimentación que obtiene al interactuar con el exterior.

Estos tres grandes grupos pueden observarse separados visualmente con mayor claridad en la figura 7.3.1, sin embargo, para efectos de esta investigación se hará énfasis en el aprendizaje supervisado, pues posteriormente se observará con claridad el problema que se trata de resolver y las características del mismo, de modo que pueden percibirse rasgos de clasificación y predicción dentro del problema a resolver; es por esto que se eligió este campo para la solución del problema.

7.3.4. Aprendizaje supervisado

El aprendizaje supervisado es un método de análisis de datos muy utilizado por la comunidad científica y que trasciende igualmente al ámbito analítico práctico; dentro de esta rama del machine learning pueden encontrarse algoritmos que "que aprenden iterativamente de los datos para permitir que los ordenadores encuentren información escondida sin tener que programar de manera explícita dónde buscar" (TIBCO Data Science, 2022).

Otros autores definen esta aproximación como "una técnica de aprendizaje automático en la que se proporciona al sistema un conjunto de datos etiquetados, con el objetivo de que el sistema aprenda a generalizar y hacer predicciones sobre nuevos datos" (Alpaydin, 2010).

Asimismo, este concepto tiene atribuciones como englobar "una clase de algoritmos de aprendizaje automático en los que se proporciona al sistema un conjunto de datos con las respuestas correctas, con el objetivo de que el sistema aprenda a hacer predicciones precisas sobre nuevos datos" (Bishop, 2006).

Para todos estos autores, este enfoque para la resolución de problemas converge en una similitud, siendo ésta la utilización de un conjunto de datos etiquetados – como se mencionó en la sección previa – de modo que se entrene a un algoritmo para una o varias tareas específicas. TIBCO Data Science (2022) usan como ejemplo en su artículo la clasificación y predicción de

imágenes en donde se perciban vehículos de dos y cuatro ruedas, donde ellos mencionan que para realizar este procedimiento deberá de entrenarse a un algoritmo con un conjunto de datos que esté correctamente etiquetado, de modo que el algoritmo debe verificar si un vehículo es de dos o de cuatro ruedas.

El aprendizaje supervisado permite “que los algoritmos ‘aprendan’ de datos históricos/de entrenamiento y los apliquen a entradas desconocidas para obtener la salida correcta. Para funcionar, el aprendizaje supervisado utiliza árboles de decisión, random forest y gradient boosting machine” (TIBCO Data Science, 2022).

Un tipo de uso muy común que se le da a nivel práctico al aprendizaje supervisado es al estudio del comportamiento en distintas compañías, creando sistemas de negocio capaces de hacer predicciones variadas, generalmente buscando optimizar las ganancias de este negocio por medio de un proceso de obtención y procesamiento de datos con estos algoritmos. Este proceso se puede ver reflejado en la figura 7.3.2.

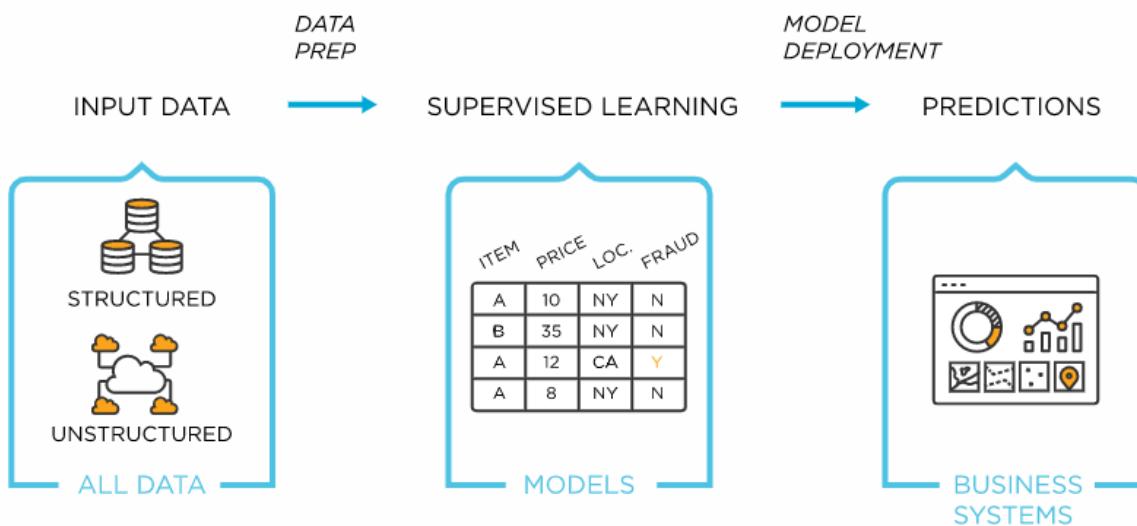


Figura 7.3.2. Procedimiento del aprendizaje supervisado (TIBCO Data Science, 2022).

Es importante aclarar que, aunque los autores previos no mencionan una característica importante del aprendizaje supervisado, ésta se encuentra intrínseca en las dos subespecializaciones que existen en este campo, tal como se menciona en la definición siguiente, acotando este proceso como uno en el que “un algoritmo de aprendizaje automático se entrena con un conjunto de datos de entrenamiento etiquetados para poder realizar tareas de clasificación o regresión” (Murphy, 2012).

Murphy en su obra otorga clarificaciones de estos dos tipos de tareas que se encuentran como subcampos de este tipo de Aprendizaje: la clasificación y la regresión; entre lo más destacable que él menciona al respecto es la comparación entre ambas, denotando que “la clasificación se

refiere a la tarea de asignar una etiqueta o categoría a una entrada, mientras que la regresión se refiere a la tarea de predecir un valor numérico continuo” (Murphy, 2012).

Aunque como se mencionó anteriormente, los demás autores no se rezagan en hacer notar esta importante distinción, pues Alpaydin (2010) también aporta su propio enunciado, donde define que “la clasificación se refiere a la tarea de asignar una etiqueta o categoría a una entrada, mientras que la regresión se refiere a la tarea de predecir un valor numérico continuo”. Por otra parte, Bishop (2006) menciona que “la clasificación es el proceso de asignar una etiqueta a una instancia, mientras que la regresión es el proceso de predecir un valor numérico continuo”.

Queda claro, entonces, que al tener en cuenta a los autores previos, la diferencia entre clasificación y regresión radica en el tipo de resultado que se desea, lo que causará que se deseé inclinarse más por un tipo de tarea u otro.

A través de las definiciones anteriores, puede construirse un concepto más apropiado para los efectos de esta investigación, pues puede observarse que el uso de clasificación recae en situaciones donde se requiere como resultado una clase de entre un número limitado de éstas, mientras que la regresión se utiliza en problemas donde el resultado será un número dentro de un conjunto infinito de posibles resultados.

Martínez Heras (2020) expone en su artículo algunos ejemplos de clasificación, entre los cuales se encuentran:

- 1) ¿Comprará el cliente este producto? [sí, no]
- 2) ¿Tipo de tumor? [maligno, benigno]
- 3) ¿Subirá el índice bursátil? IBEX mañana [sí, no]
- 4) ¿Es este comportamiento una anomalía? [sí, no]
- 5) ¿Nos devolverá este cliente un crédito? [sí, no]
- 6) ¿Qué deporte estás haciendo? Tal y como lo detectan los relojes inteligentes [caminar, correr, bicicleta, nadar]

Este mismo autor también ofrece algunos ejemplos en donde se observa el uso de regresión:

- 1) Predecir por cuánto se va a vender una propiedad inmobiliaria
- 2) Predecir cuánto tiempo va a permanecer un empleado en una empresa
- 3) Estimar cuánto tiempo va a tardar un vehículo en llegar a su destino
- 4) Estimar cuántos productos se van a vender

Finalmente, se ha visto el conocimiento suficiente para poder realizar un acercamiento a las herramientas propias de esta investigación: las técnicas de machine learning. Martínez Heras (2020) menciona como ejemplos de algunas técnicas de aprendizaje automático para la realización de clasificación y regresión son las siguientes:

- Máquinas de vectores de soporte (support vector machines).
- Árboles de decisión (decision trees).
- Bosques aleatorios (random forests).
- Redes neuronales y aprendizaje profundo (neural networks and deep learning).

Cabe aclarar que el autor menciona que, pese a que hay técnicas sumamente especializadas en clasificación y otras en regresión, la mayoría de las técnicas funcionan para ambos enfoques.

En la figura 7.3.3 se observa la separación entre los tres tipos de aprendizaje del machine learning incluyendo algunos ejemplos de técnicas que estos tipos de aprendizaje utilizan para abordar los problemas que se les presentan.

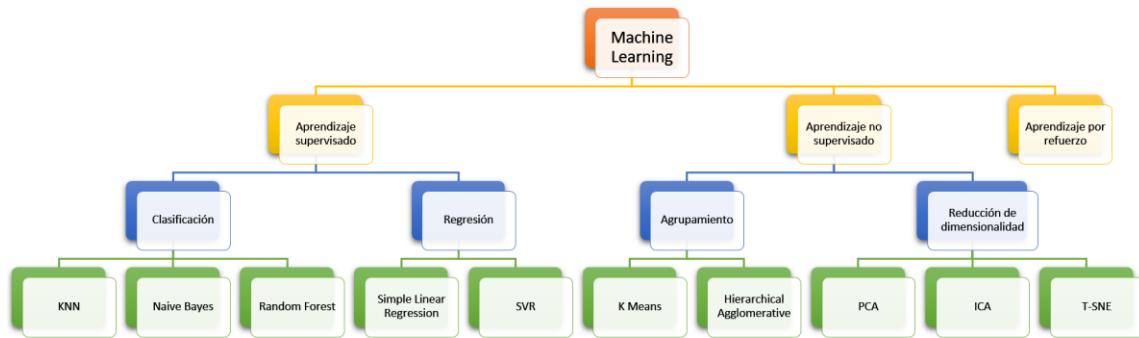


Figura 7.3.3. Principales algoritmos de machine learning (Betanzos Gómez, 2020).

Se le pide al lector dar otro salto de fe en este punto, confiando en que, por el momento se menciona que el problema de la presente investigación se trata de un problema de clasificación. Al finalizar este capítulo todos los cabos sueltos coinciden en un mismo rumbo y en el capítulo siguiente todo se habrá esclarecido – aunque con los tópicos ya revisados de electroencefalografía y aprendizaje supervisado, ya puede ser clara la idea del rumbo de esta investigación – pues todo el conocimiento de base necesario estará puesto sobre la mesa.

Dado que el problema de la presente investigación se trata de un problema de clasificación, se tomó la decisión de abordar esta pregunta de investigación con tres distintas técnicas de machine learning para clasificación: la técnica de support vector machine, la técnica de random forest y la técnica de naive Bayes. A continuación, se aclara cada técnica y se explica por qué se eligieron estas tres técnicas especialmente.

7.3.4.1. La técnica support vector machine

Esta técnica será la primera en uso en esta investigación. La máquina de vectores de soporte, o en inglés “support vector machine” (abreviada de ahora en adelante como SVM) es “un algoritmo de aprendizaje supervisado que se utiliza en muchos problemas de clasificación y

regresión, incluidas aplicaciones médicas de procesamiento de señales, procesamiento del lenguaje natural y reconocimiento de imágenes y voz" (The MathWorks Inc., s.f.).

El equipo de MathWorks (s.f.) menciona que este algoritmo trata de "encontrar un hiperplano que separe de la mejor forma posible dos clases diferentes de puntos de datos. 'De la mejor forma posible' implica el hiperplano con el margen más amplio entre las dos clases, representado por los signos más y menos en la siguiente figura".

El margen que menciona el reconocido equipo dueño de MATLAB es aquel que se define como la anchura máxima de la región paralela al hiperplano que no tiene puntos de datos interiores. El algoritmo solo puede encontrar este hiperplano en problemas que permiten separación lineal; en la mayoría de los problemas prácticos, el algoritmo maximiza el margen flexible permitiendo un pequeño número de clasificaciones erróneas.

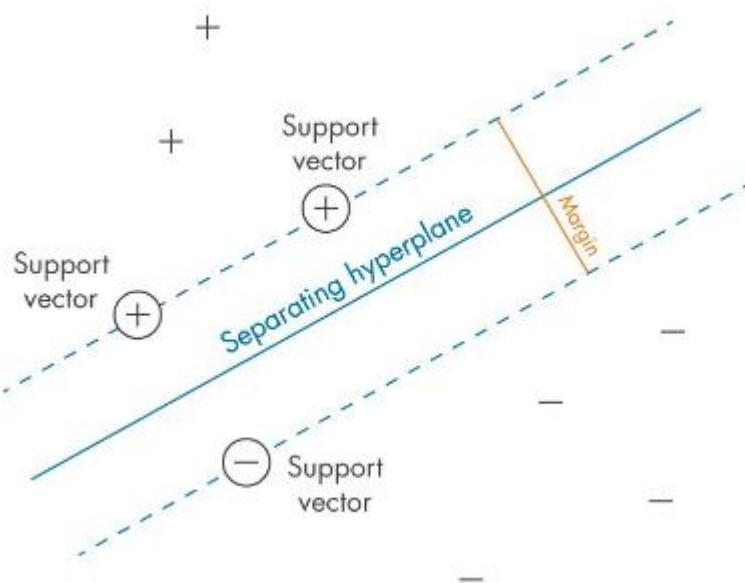


Figura 7.3.4. Definición del "margen" entre clases: el criterio que los SVM intentan optimizar (The MathWorks Inc., s.f.).

Estos vectores de soporte hacen referencia a un subconjunto de observaciones de entrenamiento, las cuales identifican posiciones del hiperplano de separación. Puede observarse una ejemplificación en la figura 7.3.4.

Un algoritmo estándar SVM está formulado para problemas de clasificación binaria, en cuanto a los problemas multiclase normalmente se reducen a una serie de problemas binarios. Existen autores que realizan esta reducción usando una técnica denominada "One vs All" (o "One vs Rest") en la cual se entrena una SVM para cada clase del problema – como lo menciona Band (2020) – comparando cada clase con las demás. Luego, se utiliza la clase con la mayor probabilidad de pertenencia para clasificar una nueva instancia. Otros autores – como

destaca Brownlee (2021) – reducen el problema mediante una técnica “One vs One” en la cual se entrena una SVM para cada par de clases, y se elige la clase con la mayor cantidad de victorias.

Cabe destacar que las SVM pertenecen a una clase de algoritmos del machine learning que son denominados “Métodos Kernel”. Estos algoritmos pueden utilizar una función de kernel para transformar las características.

Como lo menciona el equipo de MathWorks (s.f.) “las funciones de kernel asignan los datos a un espacio dimensional diferente, que suele ser superior, con la expectativa de que resulte más fácil separar las clases después de esta transformación, simplificando potencialmente los límites de decisión complejos no lineales para hacerlos lineales en el espacio dimensional de características superior asignado”. Los datos no se tienen que transformar explícitamente durante este procedimiento, lo que supondría una alta carga computacional. A esto se le conoce como “*truco de kernel*”.

Existen algunos kernels muy conocidos con los que se pueden trabajar, el equipo desarrollador de MATLAB destaca dentro de las paqueteterías de su software kernels como la función de base radial o gaussiana (véase la ecuación 7.3.1), la función lineal (véase la ecuación 7.3.2), la función polinómica (véase la ecuación 7.3.3) o la función sigmoide (véase la ecuación 7.3.4).

$$K(x_1, x_2) = \exp\left(-\frac{\|x_1 - x_2\|^2}{2\sigma^2}\right)$$

Ecuación 7.3.1. Función de base radial o gaussiana
(The MathWorks Inc., s.f.).

$$K(x_1, x_2) = x_1^\top x_2$$

Ecuación 7.3.2. Función lineal (The
MathWorks Inc., s.f.).

$$K(x_1, x_2) = (x_1^\top x_2 + 1)^\rho$$

Ecuación 7.3.3. Función polinómica (The MathWorks
Inc., s.f.).

$$K(x_1, x_2) = \tanh(\beta_0 x_1^\top x_2 + \beta_1)$$

Ecuación 7.3.4. Función sigmoide (The MathWorks
Inc., s.f.).

Un entendimiento visual que es más claro de cómo se realiza la separación del hiperplano puede observarse en la figura 7.3.5.

El entrenamiento de una SVM se muy similar a resolver un problema de optimización cuadrática para ajustar un hiperplano que minimice el margen flexible entre las clases. El número de características está determinado por el número de vectores de soporte.

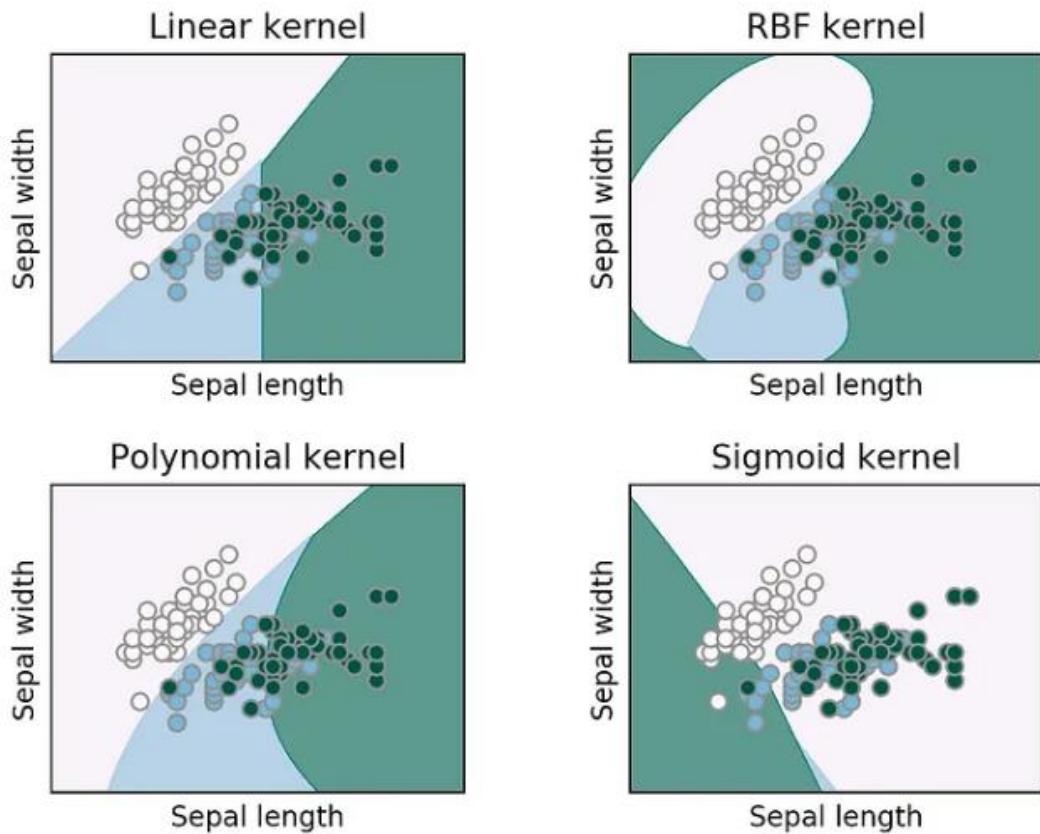


Figura 7.3.5. Gráficas de separación del hiperplano de los distintos kernels (Marius, 2020).

El uso de las SVM trae consigo ciertas ventajas, como las menciona Blog Unipython (2018):

- Es eficaz en espacios de grandes dimensiones.
- Mantiene eficacia en casos donde el número de dimensiones es mayor que el número de muestras.
- Utiliza un subconjunto de puntos de entrenamiento en la función de decisión (llamada vectores de soporte), por lo que también es eficiente en memoria.
- Es versátil dado que se pueden especificar distintas funciones del núcleo para la función de decisión. Se proporcionan kernels comunes pero también es posible especificar kernels personalizados.

Algunos puntos clave de las SVM son los siguientes según MathWorks (s.f.):

- Son muy populares y logran un buen rendimiento en muchas tareas de clasificación y regresión.
- Aunque los algoritmos SVM están formulados para la clasificación binaria, los algoritmos SVM multiclas se construyen combinando varios clasificadores binarios.

- Los kernels hacen que los SVM sean más flexibles y capaces de gestionar problemas no lineales.
- Para construir la superficie de decisión, sólo se requieren los vectores de soporte seleccionados a partir de los datos de entrenamiento. Una vez terminado el entrenamiento, el resto de datos de entrenamiento es irrelevante, produciendo una representación compacta del modelo que es adecuada para generar código de forma automatizada.

Hay que tomar en cuenta dos factores importantes “si el número de características es mucho mayor que el número de muestras evite el exceso de ajuste al elegir las funciones del Kernel y el término de regularización es crucial” (Blog Unipython, 2018). Estos mismos autores también mencionan que los SVMs no proporcionan directamente estimaciones de probabilidad, éstas se calculan utilizando una validación cruzada quíntuple.

7.3.4.2. La técnica random forest

De la siguiente técnica que se hablará será la técnica de bosques aleatorios, o en inglés “random forest”. Esta técnica es muy popular en el machine learning y el análisis de datos.

La técnica se sustenta en los llamados “árboles de decisión”; éstos “ayudan al científico de datos a tomar una decisión gracias a una serie de preguntas (también llamadas tests) cuya respuesta (sí o no) llevará a la decisión final” (Data Scientest, 2022). Pueden encontrarse ejemplos de estos algoritmos basándose en temas de Economía gracias a las facilidades que en este campo han encontrado al utilizar este algoritmo. Un ejemplo es el que brinda Sanabria Castro (2020) el cual se ilustra a continuación en la figura 7.3.6.

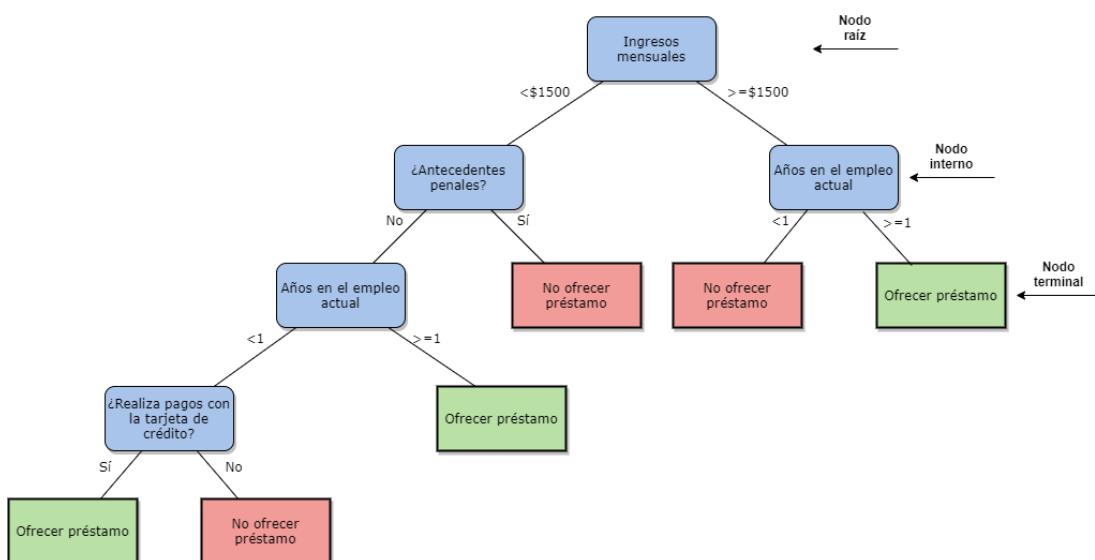


Figura 7.3.6. Árbol de Decisión aplicado en Economía (Sanabria Castro, 2020).

Como breve contextualización de los Árboles de Decisión, Sanabria Castro menciona que este tipo de modelos están formados por “una serie de decisiones lógicas similares a las de un diagrama de flujo. El árbol comienza con un nodo llamado ‘nodo raíz’, en este y en los nodos internos se toman las decisiones basado en diferentes atributos, las ramas indican las decisiones tomadas. Al final del árbol de decisión se encuentran los nodos terminales, que representan el resultado de seguir una combinación de decisiones. Alternativamente, el nodo terminal puede tener asociada una probabilidad de que el valor meta tome cierto valor” (Sanabria Castro, 2020).

Gracias a la construcción del árbol, estos modelos pueden utilizarse para la predicción de variables categóricas y continuas, recibiendo el nombre de árboles clasificación y regresión respectivamente.

La figura 7.3.6 es parte de un ejemplo que Sanabria Castro utiliza para visualizar la idea de un Árbol de Decisión mediante un ejemplo de un banco el cual utiliza un árbol de decisión para decidir si le debería ofrecer un préstamo a una persona.

Sanabria Castro (2020) menciona que “para construir un árbol de decisión es un método heurístico llamado partición recursiva. La idea es que conforme se avance en el árbol de decisión se formen grupos cada vez más uniformes, en donde predomine una clase o un valor. En el ejemplo anterior estos grupos son; personas a las que fue acertado, ofrecerles un préstamo y las que no”. Esta es la forma en la que el algoritmo encuentra patrones que son utilizados para las predicciones que se realizarán con aplicantes nuevos.

La construcción de uno se realiza mediante métodos como el ID3, el C4.5, C5.0 y el CART, sin embargo la explicación en el artículo de Sanabria Castro posee a profundidad los detalles minuciosos de estos algoritmos. La cuestión importante para efectos de esta investigación es conocer que el método está cimentado en encontrar los criterios de separación (o “Splitting Criteria”) necesarios con los que se construye el árbol.

Esta técnica presenta muchas ventajas en comparación con otros algoritmos de datos pues es fácil de interpretar, estable y generalmente regresa buenas coincidencias muy útiles en tareas de regresión o clasificación.

Enlazando la base de los árboles de decisión se obtiene el método de random forest, pues este método es considerado como un método de conjunto (o ensemble method) pues combina resultados con el objetivo de obtener un superresultado final que fue generado de los resultados de los diferentes árboles de decisión que componen al algoritmo de random forest.

Un modelo de random forest puede “constar de varias decenas, incluso centenas de árboles, el número de árboles es un parámetro que por lo general, se ajusta mediante validación

cruzada (o cross-validation, en inglés). Para abreviar, la validación cruzada es una técnica de evaluación de un algoritmo de machine learning que consiste en entrenar y probar el modelo en fragmentos de la serie de datos de partida” (Data Scientest, 2022).

El sustento del algoritmo está en entrenar a cada árbol en un subconjunto de la serie de datos y así obtener un resultado. “Posteriormente, se combinan los resultados de todos los árboles de decisión para dar una respuesta final. Cada árbol “vota” (sí o no) y la respuesta final es la que tenga la mayoría de votos” (Data Scientest, 2022).

Esto anteriormente descrito es a lo que en el artículo de Data Scientest (2022) le denominan como “bagging” y otras fuentes como el artículo en El Estadístico Blogspot (2021) coinciden en el uso de este método, el cual es un procedimiento constituido por los siguientes pasos:

- 1) Se divide la serie de datos en varios subconjuntos compuestos aleatoriamente de muestras, de ahí el “random” de random forest.
- 2) Se entrena un modelo en cada subconjunto: habrá tantos modelos como subconjuntos.
- 3) Se combinan todos los resultados de los modelos (con un sistema de voto, por ejemplo) lo que proporciona un resultado final.

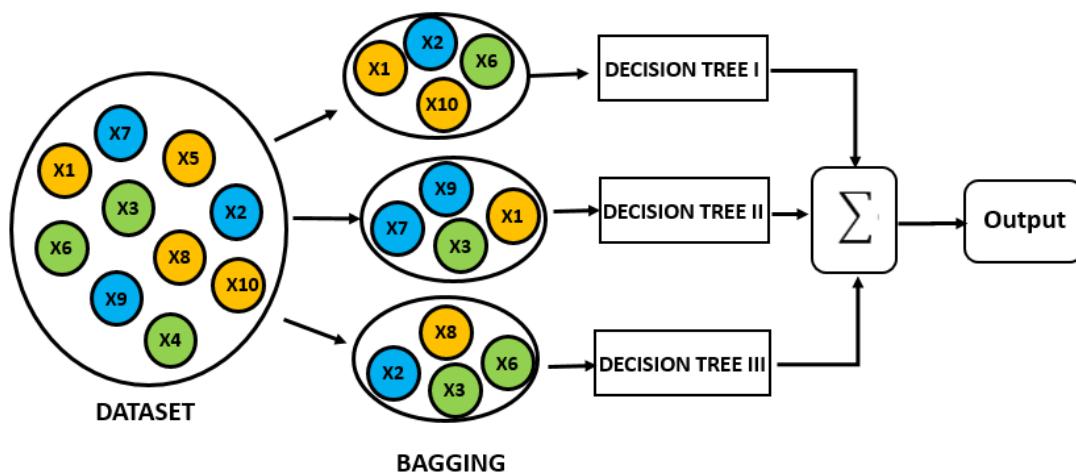


Figura 7.3.7. Visualización del “bagging” como separación, usado en random forest (Orellana Alvear, 2018).

La descripción visual del “bagging” puede observarse en la figura 7.3.7. Existe otra técnica llamada “boosting” la cual consiste en una separación y evaluación secuencial, mientras que el “bagging” puede verse como una separación y evaluación en paralelo; esta exemplificación se observa en la figura 7.3.8 pero no se le profundizará mucho más en este método, pues no es de relevancia para el presente estudio.

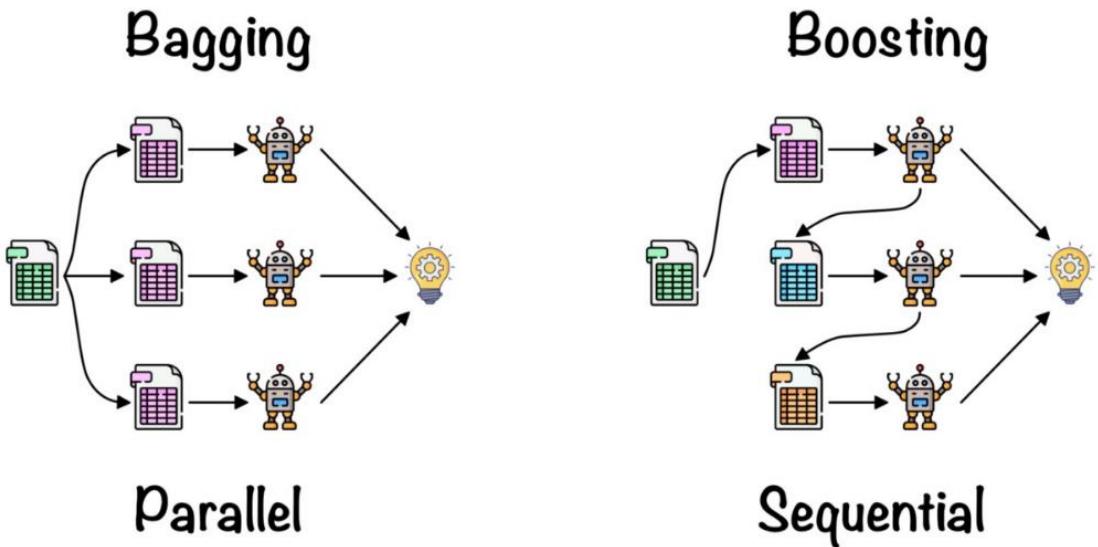


Figura 7.3.8. Comparación visual entre el “bagging” y el “boosting” (Sruthi, 2022).

Matemáticamente, el funcionamiento del random forest se puede dividir en cuatro pasos según El Estadístico Blogspot (2021):

- I. Seleccionamos k *features* o características (columnas o variables) de las m totales (siendo $k < m$) y se crea un árbol de decisión con esas k características.
- II. Se crean n árboles variando siempre la cantidad de k *features* o características. También podría variarse la cantidad de muestras que se pasan a esos árboles (esto es conocido como “*bootstrap sample*”).
- III. Se toma cada uno de los n árboles y se ejecutan realizando una misma clasificación. Se guarda el resultado de cada árbol obteniendo n salidas.
- IV. Se calculan los votos obtenidos para cada clase seleccionada y se considerará a la más votada como la clasificación final de nuestro bosque.

7.3.4.3. La técnica naive Bayes

Como última técnica que se abordará bajo la lupa de esta investigación se tiene a la técnica de Bayes ingenuo, o en inglés “naive Bayes”. Estos modelos son una clase especial de algoritmos de clasificación dentro del aprendizaje automático y están basados en una técnica de clasificación estadística llamada el teorema de Bayes.

Para brindar superficialmente el conocimiento del teorema de Bayes al lector se resalta que éste mismo es el núcleo de la inferencia Bayesiana; un campo muy amplio dentro de la teoría de la probabilidad. Se trata de una proposición planteada por el matemático inglés Thomas Bayes la cual fue publicada en 1763 tal como se resalta en el ensayo que el mismo Bayes

(1763) publicó. En términos simples, en este ensayo se encuentra este teorema el cual expresa la probabilidad condicional de un evento aleatorio A dado B en términos de la distribución de probabilidad condicional del evento B dado A y la distribución de probabilidad marginal de solo A .

El teorema interpretado por Parzen (1987) está enunciado de la siguiente forma:

Sea $\{A_1, A_2, \dots, A_i, \dots, A_n\}$ un conjunto de sucesos mutuamente excluyentes y exhaustivos tales que la probabilidad de cada uno de ellos es distinta de cero: ($P[A_i] \neq 0$ para $i = 1, 2, \dots, n$). Si B es un suceso cualquiera del que se conocen las probabilidades condicionales $P(B|A_i)$, entonces la probabilidad $P(A_i|B)$ viene dada por la expresión:

$$P(A_i|B) = \frac{P(B|A_i)P(A_i)}{P(B)}$$
Ecuación 7.3.5. El teorema de Bayes (Parzen, 1987).

Donde:

- $P(A_i)$ son las probabilidades a priori.
- $P(B|A_i)$ es la probabilidad de B en la hipótesis A_i .
- $P(A_i|B)$ son las probabilidades resultantes.

El teorema de Bayes puede visualizarse al superponer dos árboles de decisión como se observa en la figura 7.3.9.

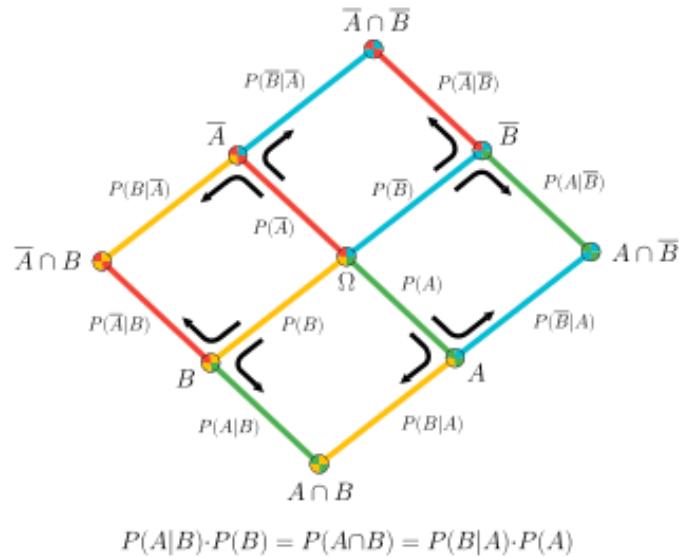


Figura 7.3.9. Visualización del teorema de Bayes por superposición de dos árboles de decisión (Parrás & Tedesco).

Ahora, con base en la definición de probabilidad condicionada se obtiene la fórmula de Bayes. También se le llama regla de Bayes como Walpole, Myers, Myers y Ye (2012) la denominan.

Ellos la definen matemáticamente en la obra que ellos escribieron titulada “*Probabilidad y Estadística para Ingeniería y Ciencias*” de la siguiente manera:

$$P(A_i|B) = \frac{P(B|A_i)P(A_i)}{\sum_{k=1}^n P(B|A_k)P(A_k)}$$

Ecuación 7.3.6. La regla de Bayes (Walpole, Myers, Myers, & Ye, 2012).

La teoría estadística previa es la base de esta técnica, además de la característica de estos modelos de ser “Naive” o ingenuos en español. Se les llama así debido a que este tipo de esquemas asume que las variables predictoras son independientes entre sí.

Debido a la simplicidad de esta técnica, proporcionan una manera muy sencilla de elaborar modelos con un comportamiento óptimo. Esto lo consiguen, según Roman (2019) “proporcionando una forma de calcular la probabilidad ‘posterior’ de que ocurra un cierto evento A, dadas algunas probabilidades de eventos ‘anteriores’”. Matemáticamente se expresa como se observa en la figura 7.3.10.

$$P(A|R) = \frac{P(R|A)P(A)}{P(R)}$$

$P(A)$: Probabilidad de A

$P(R|A)$: Probabilidad de que se de R dado A

$P(R)$: Probabilidad de R

$P(A|R)$: Probabilidad posterior de que se de A dado R

Figura 7.3.10. El uso del teorema de Bayes en la técnica de naive Bayes (Roman, 2019).

Para poder utilizar esta técnica en problemas de clasificación, Roman (2019) menciona los siguientes pasos necesarios para el trabajo con este algoritmo:

- 1) Convertir el conjunto de datos en una tabla de frecuencias.
- 2) Crear una tabla de probabilidad calculando las correspondientes a que ocurran los diversos eventos.
- 3) La ecuación naive Bayes se usa para calcular la probabilidad posterior de cada clase.
- 4) La clase con la probabilidad posterior más alta será el resultado de la predicción.

Esta técnica tiene ciertas ventajas y desventajas de su uso. Entre las ventajas que destaca Roman (2019) se encuentran las siguientes:

- I. Un manera fácil y rápida de predecir clases, para problemas de clasificación binarios y multiclase.
- II. En los casos en que sea apropiada una presunción de independencia, el algoritmo se comporta mejor que otros modelos de clasificación, incluso con menos datos de entrenamiento.

III. El desacoplamiento de las distribuciones de características condicionales de clase significa que cada distribución puede ser estimada independientemente como si tuviera una sola dimensión. Esto ayuda con problemas derivados de la dimensionalidad y mejora el rendimiento.

Por otra parte, Roman (2019) señala las siguientes desventajas:

- a. Aunque son unos clasificadores bastante buenos, los algoritmos naive Bayes son conocidos por ser pobres estimadores. Por ello, no se deben tomar muy en serio las probabilidades que se obtienen.
- b. La presunción de independencia naive muy probablemente no reflejará cómo son los datos en el mundo real.
- c. Cuando el conjunto de datos de prueba tiene una característica que no ha sido observada en el conjunto de entrenamiento, el modelo le asignará una probabilidad de cero y será inútil realizar predicciones. Uno de los principales métodos para evitar esto, es la técnica de suavizado, siendo la estimación de Laplace una de las más populares.

Estos serán los algoritmos que se utilizarán posteriormente. Con esto, queda cubierto el conocimiento requerido dentro del campo de las ciencias de la computación; antes de cerrar el capítulo hay que hablar de cómo introducir el conjunto de datos a estos algoritmos; procedimiento que se define como el preprocesamiento de datos.

7.3.5. Un poco de matemáticas en el preprocesamiento de datos

En esta investigación se plantea cuál de los tres algoritmos mencionados previamente es el mejor clasificador dentro de un ámbito de intención del movimiento, no obstante, un clasificador también debe su rendimiento en gran medida a las técnicas de preprocesamiento de datos que se usen sobre los conjuntos de datos previos a insertar estos mismos dentro de los algoritmos mencionados.

Lo anterior se apoya debido a que el preprocesamiento de datos “se vislumbra como una herramienta muy importante en el paso de big data a smart data, esencial para convertir los datos almacenados (material en bruto) en datos de calidad” (Salvador, Ramírez-Gallego, Luengo, & Herrera, 2014).

Salvador y otros (2014) hablan del preprocesamiento de datos como una etapa esencial en el proceso de descubrimiento de información en bases de datos o KDD por sus siglas en inglés “*knowledge discovery in databases*” considerándose incluso como de uso obligado por ellos mismos, pues sin ellas, algoritmos de extracción de conocimiento no podrían ejecutarse correctamente o los resultados que regresen serían erróneos.

Para la presente investigación se trataron dos técnicas de preprocesamiento de datos; de estas dos técnicas se abordarán sus aspectos técnicos abordando un poco de matemáticas en este apartado. Las técnicas que se usaron dentro de los conjuntos de datos que se exponen a continuación son un preprocesamiento de normalización sobre los datos en bruto conocido como normalización Z. Posteriormente, sobre los datos obtenidos tras este procedimiento se realizó un segundo preprocesamiento, ahora de transformación usando la transformada de Fourier bajo un enfoque de magnitud y fase.

La implementación de ambos tipos de preprocesamiento no compete al apartado actual, pues se explica en capítulos sucesivos, pero a continuación se aborda a nivel teórico de qué trata cada uno de estos métodos de preprocesamiento de datos.

7.3.5.1. Normalización z

Un procedimiento de normalización sobre un conjunto de datos es conveniente pues realiza ajustes iniciales en el conjunto que permiten una mejora al usar este conjunto de datos como entrada en algoritmos de clasificación. Propiamente este tipo de normalización parte de la base estadística usando el concepto de la distribución normal.

En estadística, la distribución normal estándar es una función de densidad de probabilidad, probablemente de las más importantes en el campo de la estadística y probabilidad. Ésta “trata de una discusión sobre la probabilidad y, por tanto, son los datos de la población los que pueden estar distribuidos normalmente, y si lo están, entonces es así como podemos calcular las probabilidades de eventos específicos” (Holmes, Illowsky, & Dean, 2022).

Si un fenómeno se considera que puede tener una distribución de probabilidad normal, la forma de calcular eventos probables dentro de este fenómeno se reduce al uso de “dos parámetros (dos medidas numéricas descriptivas): la media (μ) y la desviación típica (σ). Si X es una cantidad que se va a medir que tiene una distribución normal con media (μ) y desviación típica (σ)” (Holmes, Illowsky, & Dean, 2022).

$$f(x) = \frac{1}{\sigma \cdot \sqrt{2 \cdot \pi}} \cdot e^{-\frac{1}{2} \left(\frac{x-\mu}{\sigma} \right)^2}$$

Ecuación 7.3.7. Función de densidad de probabilidad de una distribución normal (Holmes, Illowsky, & Dean, 2022).

Holmes, Illowsky y Dean (2022) declaran la ecuación 7.3.7 como la función de densidad de probabilidad de una distribución normal. Ellos también mencionan ciertas características muy peculiares de esta distribución de probabilidad, hechos como observar que “la curva es simétrica respecto a una línea vertical que pasa por la media, μ . La media es la misma que la mediana, que es la misma que la moda, porque el gráfico es simétrico respecto a μ . Como

indica la notación, la distribución normal solo depende de la media y de la desviación típica” (Holmes, Illowsky, & Dean, 2022).

Estas declaraciones de los tres autores también las ilustran en un gráfico de la función de distribución normal, el cual en este documento puede apreciarse en la figura 7.3.11 donde se observan las características recientemente mencionadas de la distribución de forma visual.

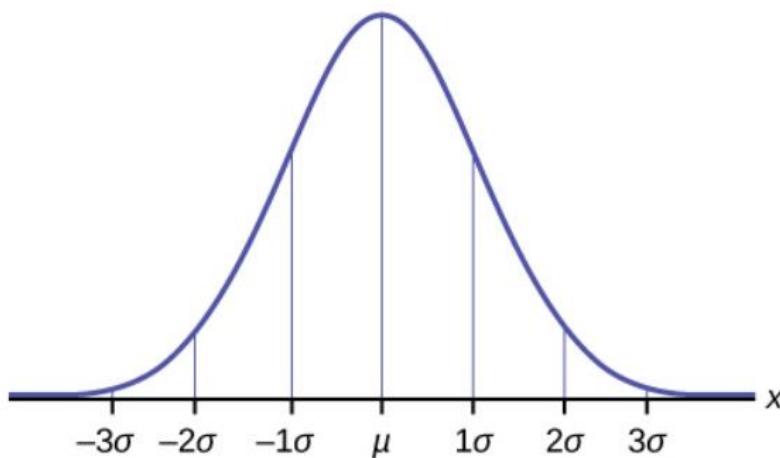


Figura 7.3.11. Gráfica de la función de una distribución normal (Holmes, Illowsky, & Dean, 2022).

Algunas otras peculiaridades de la distribución normal que los tres autores comentan son cuestiones como observar que, en una distribución normal estándar, la media es cero y la desviación típica es uno, permitiendo simplificar el cálculo matemático de las probabilidades.

Llevar a estos valores una distribución normal es lo que se le conoce como “estandarizar” la distribución, comentan Holmes, Illowsky y Dean (2022), pues gracias a la ecuación 7.3.8 obtenemos la distribución $Z \sim N(0,1)$.

$$z = \frac{x - \mu}{\sigma}$$

Ecuación 7.3.8. Transformación para obtener las puntuaciones z (Holmes, Illowsky, & Dean, 2022).

Es con la ecuación 7.3.8 con la que puede convertirse un problema donde, si X es una variable aleatoria normalmente distribuida y $X \sim N(\mu, \sigma)$, entonces el uso de la ecuación 7.3.8 para obtener la puntuación z para una determinada x permite obtener la ya mencionada $Z \sim N(0,1)$.

De este modo, esta ecuación “indica cuántas desviaciones típicas tiene el valor x por encima (a la derecha) o por debajo (a la izquierda) de la media, μ . Los valores de x que son mayores que la media, tienen puntuaciones z positivas, y los valores de x que son menores que la media, tienen puntuaciones z negativas. Si x es igual a la media, entonces x tiene una puntuación z de cero” (Holmes, Illowsky, & Dean, 2022).

Esta es la base de la normalización z. Pues el objetivo será tratar al conjunto de datos como una distribución normal. Partiendo de esta base, el conjunto de datos será reestructurado a valores de una distribución $Z \sim N(0,1)$. Es así como se haría el primer preprocesamiento que posteriormente se muestra en etapas de experimentación y desarrollo.

Sin embargo no es la única etapa que existe expuesta en este documento. Antes de introducir el conjunto de datos a los algoritmos previamente mencionados también se realizó un procedimiento de transformación utilizando la transformada de Fourier el cual a nivel teórico se expone a continuación.

7.3.5.2. Transformación de Fourier

Este subapartado quizá contiene las matemáticas más complejas del documento, pero se espera personalmente explicarlas con facilidad pues no se profundiza mucho en su aplicación y el trabajo no es ampliamente enfocado en este instrumento.

La transformada de Fourier es una transformación matemática que se emplea “para transformar señales entre el dominio del tiempo (o espacial) y el dominio de la frecuencia, que tiene muchas aplicaciones en la física y la ingeniería. Es reversible, siendo capaz de transformarse en cualquiera de los dominios al otro. El propio término se refiere tanto a la operación de transformación como a la función que produce” (Wikipedia, Transformada de Fourier, s.f.).

Un dato importante de esta transformación matemática es que “en el caso de una función periódica en el tiempo (no necesariamente sinusoidal), la transformada de Fourier se puede simplificar para el cálculo de un conjunto discreto de amplitudes complejas, llamado coeficientes de las series de Fourier. Ellos representan el espectro de frecuencia de la señal del dominio-tiempo original” (Wikipedia, Transformada de Fourier, s.f.). Esto es útil dado que la investigación se basa en una función en el tiempo de señales, en este caso, de los registros de actividad cerebral de un usuario.

La transformada de Fourier está definida en la ecuación 7.3.9

$$g(\xi) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} f(x)e^{-i\xi x} dx$$

Ecuación 7.3.9. La transformada de Fourier
(Wikipedia, Transformada de Fourier, s.f.).

Esta operación puede definirse dentro de un entorno cotidiano como el espectro de frecuencias de una función, similar a “lo que hace el oído humano, ya que recibe una onda auditiva y la transforma en una descomposición en distintas frecuencias (que es lo que finalmente se escucha). El oído humano va percibiendo distintas frecuencias a medida que pasa el tiempo, sin embargo, la transformada de Fourier contiene todas las frecuencias del

tiempo durante el cual existió la señal; es decir, en la transformada de Fourier se obtiene un solo espectro de frecuencias para toda la función” (Wikipedia, Transformada de Fourier, s.f.).

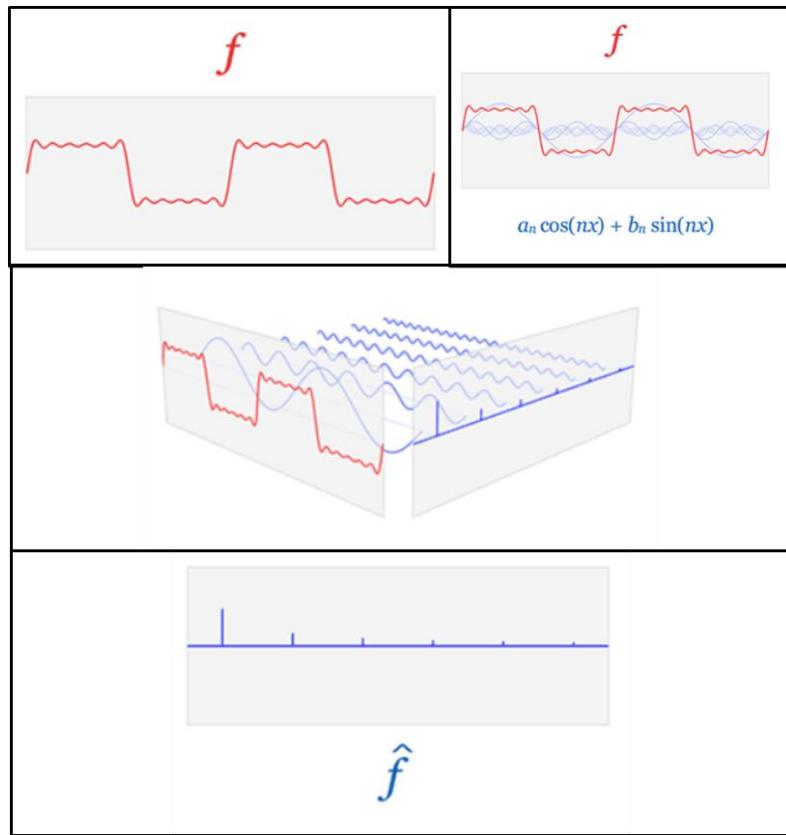


Figura 7.3.12. La transformada de Fourier visualmente (Wikipedia, Transformada de Fourier, s.f.).

Este cálculo resultante se puede simplificar en un conjunto de amplitudes complejas relatado previamente; a este conjunto se le conoce como coeficientes de las series de Fourier que son números complejos con dos formas equivalentes, la forma de coordenadas polares $Ae^{i\theta}$ y la forma de coordenadas rectangulares $A \cos(\theta) + iA \sin(\theta)$. Esta igualdad se aprecia en la ecuación 7.3.10.

$$\hat{f}(\xi) = Ae^{i\theta} = A \cos(\theta) + iA \sin(\theta)$$

Ecuación 7.3.10. El resultado de la transformada de Fourier simplificado a las dos formas equivalentes de los números complejos (Wikipedia, Transformada de Fourier, s.f.).

Muy brevemente profundizando en la experimentación – para proseguir teóricamente con la explicación – la frecuencia obtenida por cada nodo se le realiza este procedimiento obteniendo un valor complejo del tipo $a + bi$. A partir de aquí, para insertar los valores en los modelos de machine learning, antes hay que darles una forma según cierto enfoque. se deben transformar los datos complejos en características reales que puedan ser comprendidas por el modelo.

Hay diferentes enfoques para realizar esta transformación, dos de los más comunes que menciona The MathWorks Inc. (2022) son:

- **Enfoque basado en magnitud y fase:** En este enfoque, se separan los datos complejos en sus componentes de magnitud y fase, creando así dos nuevas características para cada valor complejo. La magnitud se define como la distancia del punto en el plano complejo al origen, y la fase se define como el ángulo entre el punto y el eje real. Estas dos características pueden ser utilizadas como entradas separadas para un modelo de machine learning.
- **Enfoque basado en representación de amplitud y frecuencia:** En este enfoque, se extraen los componentes de amplitud y frecuencia de los datos complejos. La amplitud representa la fuerza o intensidad de cada frecuencia en la señal, y la frecuencia representa la frecuencia en sí misma. Se puede utilizar una técnica de extracción de características como transformada de coseno discreta (DCT) para extraer estas características. Luego, se pueden utilizar como entradas para el modelo de machine learning.

El enfoque al que se le quiere dar especial atención es al enfoque basado en magnitud y fase. Se dará un ejemplo de interpretación personal a la definición previamente dada para explicar gráficamente en qué consiste este enfoque.

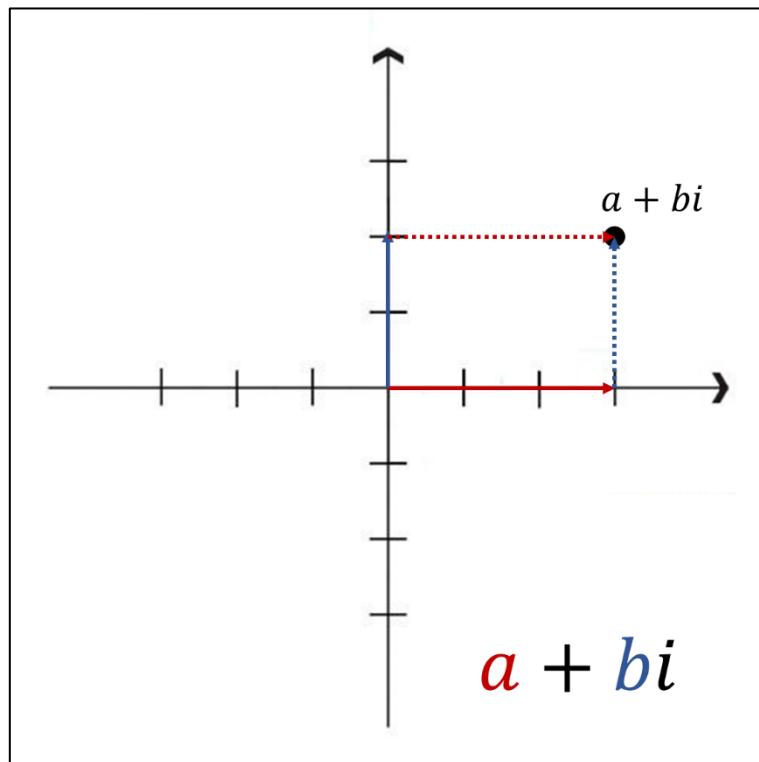


Figura 7.3.13. Representación en el plano complejo de un número complejo. Elaboración propia.

Apoyado en el artículo de Khan Academy Authors explicando el plano complejo, un número complejo de la forma $a + bi$ puede visualizarse en el plano complejo como se observa en la figura 7.3.13, donde uno de los ejes (el eje x en el plano cartesiano) corresponde a la parte real del número complejo y el otro de los ejes (el eje y en el plano cartesiano) a la parte imaginaria. Cabe señalar que el valor a del número es la parte real, mientras que el valor b es la parte imaginaria por estar acompañada de la unidad imaginaria: i que, dentro de los números complejos, cabe recordar que $i^2 = -1$ por lo que $\sqrt{-1} = i$.

Una vez concebida la figura 7.3.13, es posible dar el paso a tomar el enfoque de magnitud y fase, pues la magnitud (m) se define como la distancia del punto en el plano complejo al origen. Así también, la fase (φ) es el ángulo entre el punto y el eje real.

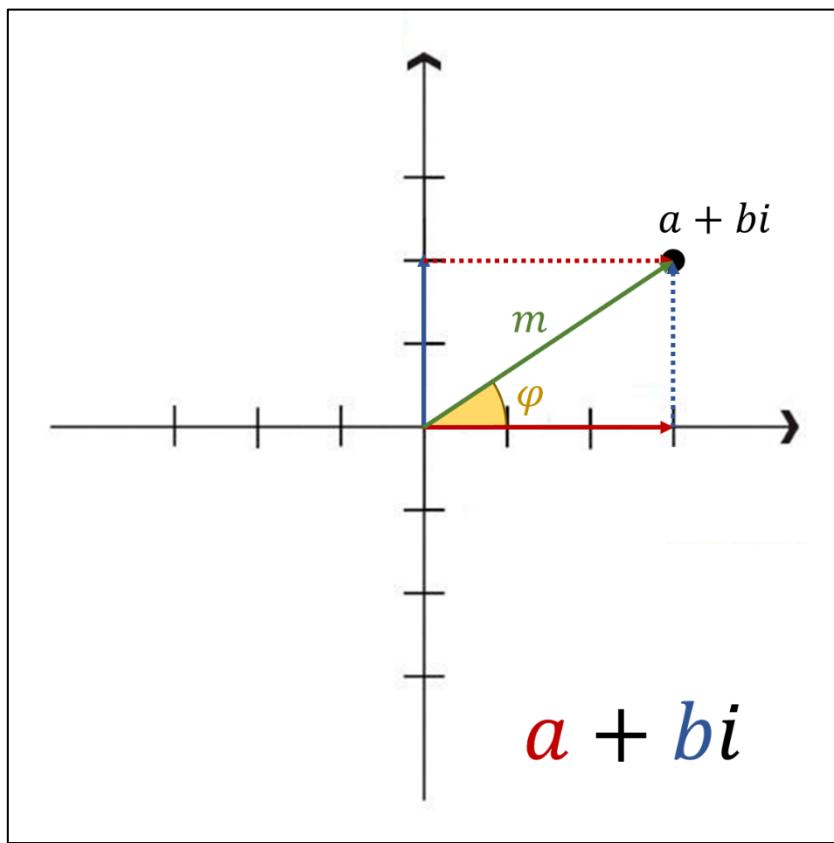


Figura 7.3.14. Representación de un número complejo bajo el enfoque de magnitud y fase. Elaboración propia.

Esto se visualiza como se aprecia en la figura 7.3.14, además que sus cálculos pueden realizarse bajo inspección matemática, pues, por ejemplo, la magnitud puede obtenerse trazando un triángulo donde los catetos son a y b y la hipotenusa sería m realizando un teorema de Pitágoras básico como el expresado en la ecuación 7.3.11.

$$m = \sqrt{a^2 + b^2}$$

Ecuación 7.3.11. Cálculo de la magnitud de un número complejo. Inferencia propia.

La fase tiende a calcularse de forma más compleja (bajo inferencia propia) pues hay que considerar casos distintos si el punto se encuentra en alguno de los cuatro distintos cuadrantes del plano complejo, sin embargo, en los campos de computación y matemáticas existe una función creada para evitar esto que es la función *atan2* o la función arcotangente de dos argumentos.

Antes de definir esta función, definiremos la función arcotangente tradicional que es de donde se origina. La función arcotangente, según Wikipedia (s.f.) en su artículo de la arcotangente, es la función inversa a la función tangente que al darle un valor θ a ésta, nos indica de forma ilustrativa, dentro de un triángulo rectángulo, la relación que existe entre el cateto opuesto al ángulo y el cateto adyacente al mismo ángulo. Esta función realiza cálculos de valores de θ para $\theta \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$.

La función arcotangente trabaja de forma inversa, de manera ilustrativa, al introducir valores de la relación de ambos catetos de un triángulo rectángulo, ésta nos proporciona el ángulo formado θ . Es de esta raíz que se origina la función *atan2* pues, según Wikipedia (s.f.) en su artículo en inglés de dicha función, la definición de esta función se presenta en la ecuación 7.3.12 y es el valor del ángulo en radianes entre el eje x y el punto dado por las coordenadas (x, y) en el plano.

$$\theta = \text{atan2}(y, x)$$

Ecuación 7.3.12. Definición de la función *atan2*.
(Wikipedia, atan2, s.f.).

La definición que se le da a esta función partiendo de la función arcotangente original puede encontrarse en la ecuación 7.3.13 que muestra lo que se exponía anteriormente que deben de considerarse ciertos casos según el cuadrante en el que se encuentre, englobando todos ellos para producir la función que nos ayudará a realizar el cálculo de la fase.

$$\text{atan2}(y, x) = \begin{cases} \arctan\left(\frac{y}{x}\right) & \text{si } x > 0 \\ \arctan\left(\frac{y}{x}\right) + \pi & \text{si } x < 0, y \geq 0 \\ \arctan\left(\frac{y}{x}\right) - \pi & \text{si } x < 0, y < 0 \\ +\frac{\pi}{2} & \text{si } x = 0, y > 0 \\ -\frac{\pi}{2} & \text{si } x = 0, y < 0 \\ \text{indefinido} & \text{si } x = 0, y = 0 \end{cases}$$

Ecuación 7.3.13. Definición de la función *atan2*
partiendo de la función arcotangente tradicional.
(Wikipedia, atan2, s.f.).

Estos cálculos se realizan al conjunto de datos obtenido en experimentación de campo, obteniendo por cada canal de frecuencia un valor m y φ que serán los definitivos a evaluarse dentro de los algoritmos previamente expuestos de machine learning.

Este es el conocimiento que abarca esta investigación dentro del campo de las ciencias de la computación; finalmente se cubrirán tópicos del tema de interés de esta investigación, el cual se tratará de la clasificación de intención del movimiento. Tema del que se hablará a profundidad en los capítulos posteriores.

7.4. Los procesos mentales de esta investigación

El pensamiento humano puede literalmente, transformar el mundo físico
Dan Brown: El Símbolo Perdido

Ha llegado el momento, pues tras todo el marco teórico previo, es el momento de abordar el problema de la presente investigación. Los temas previos funcionaron para poner sobre la mesa todo conocimiento necesario para enfocarse ahora en la clasificación de intención del movimiento.

Sin embargo, previo a esto, hay que analizar dos procesos mentales más, primeramente porque uno de ellos, el habla imaginada, posee cierta investigación que apoya a la presente exploración; por otra parte, la evocación de un concepto, pese a que se aborda vagamente en el presente, no es de interés de la investigación pues no se consiguieron resultados comparables con los de intención del movimiento.

Es así que este capítulo estará separado en estos tres procesos mentales, dando inicio al apartado con el primero de los procesos: el habla imaginada, revisando investigación ciertamente importante para el desarrollo de la presente indagación científica.

7.4.1. Un preámbulo: habla imaginada

Hay algunos documentos de investigación que profundizan en un estudio de señales electroencefalográficas con machine learning tratando de obtener información de estos datos a partir de una idea primitiva de lo que es el habla imaginada.

El habla imaginada es un campo novedoso dentro de disciplinas que tratan tópicos muy relacionados, tal como lo es la psicología. Posee distintas definiciones entre los autores que se han acercado a este ámbito. En la obra de Sheikh y Korm (1994) se da como definición del habla imaginada “la habilidad de crear imágenes mentales y asociarlas con el lenguaje” así como también se destaca que es “la capacidad de una persona de hablar en su mente sin pronunciar en voz alta las palabras” (Sheikh & Korm, 1994).

Por otra parte Singer (2006) en su obra “Imagery in Psychotherapy” define este concepto como “una habilidad cognitiva que consiste en el uso de imágenes mentales para generar palabras o frases”.

Puede referirse al habla imaginada como un término que se refiere a la capacidad de una persona de hablar en su mente sin pronunciar en voz alta las palabras. Esta habilidad puede estar relacionada con la capacidad de la mente de crear imágenes mentales y de asociarlas con el lenguaje. Sin embargo, aún no se conoce mucho sobre cómo funciona exactamente el habla imaginada y más investigación es necesaria para entenderlo mejor.

Un trato muy adecuado es el que le dan Marín, Martínez, Ureña y López (2017) en su artículo, que es al que se apega este capítulo con mayor medida. Estos autores comienzan en su artículo preguntándose sobre si somos capaces de interpretar las señales eléctricas que genera nuestro cerebro mientras estamos pensando. Ellos desde un inicio dan pie a la posibilidad de esto gracias a los avances del aprendizaje automático e incluso mencionan “nos encontramos en el inicio de una nueva era en lo que al conocimiento del cerebro se refiere” (Marín, Martínez, Ureña, & López, 2017).

Estos autores también dan su definición, clara y concisa respecto al habla imaginada: “palabras que se piensan, pero no se dicen”. Mencionan los avances que se están consiguiendo, tales como distintos resultados al proveer un vocabulario reducido consiguiendo que una persona se comunique con un equipo de cómputo a través del pensamiento, únicamente con ayuda del hardware de adquisición de la actividad cerebral. Esto fue una obra experimental que realizó el vinculado al Instituto Nacional de Astrofísica Óptica y Electrónica (INAOE) Torres-García (2013) junto a otros investigadores citados a continuación.

Este artículo incluso es citado en el artículo de Marín, Martínez, Ureña y López, siendo una base importante para los estudios del habla imaginada y también para la presente investigación. El artículo plantea su uso de las interfaces cerebro – computadora “mecanismos neurológicos o procesos empleados por el usuario para generar las señales de control, se les denomina fuentes electrofisiológicas. Las más utilizadas son: los potenciales corticales lentos (SCP, por sus siglas en inglés), los potenciales P300, las imágenes motoras (ritmos sensoriales motrices mu y beta) y los potenciales evocados visuales (VEP, por sus siglas en inglés)” (Torres-García, Reyes-García, Villaseñor-Pineda, & Ramírez-Cortés, 2013).

Dentro de su investigación definen que “la fuente electrofisiológica es el habla imaginada, también referida como habla interna o habla no pronunciada (unspoken speech), donde el término habla imaginada se refiere a la pronunciación interna, o imaginada, de palabras pero sin emitir sonidos ni articular gestos para ello” (Torres-García, Reyes-García, Villaseñor-Pineda, & Ramírez-Cortés, 2013) la cual es la brújula de la definición del habla imaginada a lo largo de toda la publicación.

Así también, mencionan citando a otra publicación académica realizada por Denby y otros (2010) mencionando que este autor “incluye a estos trabajos dentro de un área de investigación denominada interfaces de habla silente (SSI, por Silent Speech Interfaces) cuya finalidad es desarrollar sistemas capaces de permitir la comunicación "hablada" que toman lugar cuando la emisión de una señal acústica entendible es imposible” (Torres-García, Reyes-García, Villaseñor-Pineda, & Ramírez-Cortés, 2013).

Torres-García y sus colaboradores remarcan la importancia de separar los trabajos de habla imaginada en dos enfoques: palabras y sílabas. En el artículo se encuentran referencias a otras

publicaciones de ambos enfoques sirviendo como ejemplos principales de cada uno de los enfoques. Posteriormente también explica brevemente los alcances y limitantes de las publicaciones que menciona como referentes de ambas formas de tratamiento de este problema.

Una vez que termina de dar este repaso al estado del arte que tienen Torres-García y su grupo de autores colaboradores, procede a explicar que su propuesta se centra en “interpretar las señales para reconocer la pronunciación imaginada de palabras de un vocabulario reducido compuesto por las siguientes cinco palabras en lenguaje español: “arriba”, “abajo”, “izquierda”, “derecha” y “seleccionar”” (Torres-García, Reyes-García, Villaseñor-Pineda, & Ramírez-Cortés, 2013).

Para descifrar las palabras a partir de un EEG, como primer paso debe contemplarse “el registro de la actividad eléctrica cerebral como consecuencia de la activación de grupos de neuronas, lo que habitualmente se denomina un EEG. Ello requiere el uso de un hardware de uso específico que registre tal actividad. Están proliferando muchas empresas dedicadas al desarrollo de este tipo de hardware (Emotiv – emotiv.com, Neurosky – neurosky.com, GTEC – gtec.at)” (Marín, Martínez, Ureña, & López, 2017).

Estos autores mencionan que es necesario tomar en cuenta la cantidad y la posición de los electrodos pues, “si lo que pretendemos es analizar el habla imaginada, el hardware necesariamente deberá tener electrodos localizados sobre el área del cerebro relacionada con esa tarea” (Marín, Martínez, Ureña, & López, 2017).

El objetivo de la investigación señalada “retoma lo planteado por los trabajos previos con la idea en mente de que las palabras con mayor significado semántico puedan generar mayor actividad cerebral que permita su reconocimiento y correcta clasificación. Además, con ellas sería posible controlar la dirección del cursor de la computadora. El problema es tratado bajo un enfoque de clasificación, y se conoce a priori en qué parte de la señal de EEG la persona imagina la pronunciación de las palabras indicadas” (Torres-García, Reyes-García, Villaseñor-Pineda, & Ramírez-Cortés, 2013).

Como bien se explica en el artículo, al terminar la adquisición de datos, el siguiente paso es el preprocesamiento de las ondas cerebrales, que “consiste básicamente en la eliminación de ciertos artefactos (ruidos, interferencias...) con el fin de evitar cualquier tipo de injerencia no deseada en la señal de interés. Los artefactos más típicos suelen ser de origen biológico (latidos de corazón, parpadeo...) y están localizados en el mismo cuerpo del sujeto a estudiar, aunque debido a la naturaleza de estas ondas y a su rango de amplitudes, no es raro encontrar artefactos procedentes de otros lugares” (Marín, Martínez, Ureña, & López, 2017).

El método usado es el método de referencia promedio común, o CAR, por sus siglas en inglés, como se explica en el artículo de Torres-García y otros (2013). Ellos señalan que el CAR puede ser calculada mediante la siguiente fórmula:

$$V_i^{CAR} = V_i^{ER} - \frac{1}{n} \sum_{j=1}^n V_j^{ER}$$

Ecuación 7.4.1. Fórmula del cálculo del CAR (Torres-García, Reyes-García, Villaseñor-Pineda, & Ramírez-Cortés, 2013).

Donde:

- V_i^{ER} es el potencial entre el i -ésimo electrodo y la referencia.
- n es el número de electrodos en el montaje.

También se destaca que en la etapa de preprocesado también “podemos aplicar diferentes técnicas que facilitan la manipulación de los datos obtenidos mediante el EEG. Los EEG son, en definitiva, el registro de señales cuyo valor varía en cada instante temporal. De este modo es posible, por ejemplo, filtrar la señal original quedándonos solamente con las bandas de frecuencia que tienen una carga informativa mayor o que caen dentro de la frecuencia que deseamos estudiar” (Marín, Martínez, Ureña, & López, 2017); esto porque las BCI son no estacionarias puesto que las señales del EEG pueden variar rápidamente con el tiempo.

Torres-García y otros (2013) proponen como técnica que permite modelar dichas variaciones en el dominio tiempo-escala es la Transformada Wavelet Discreta (DWT por sus siglas en inglés). “La DWT provee una representación wavelet altamente eficiente mediante la restricción de la variación en la traslación y la escala, usualmente a potencias de dos. En ese caso, la DWT es algunas veces llamada transformada wavelet diádica” (Torres-García, Reyes-García, Villaseñor-Pineda, & Ramírez-Cortés, 2013).

La DWT, según Torres-García y otros (2013), se define mediante la siguiente ecuación:

$$W(j, k) = \sum_j \sum_k f(x) 2^{-\frac{j}{2}} \psi(2^{-j}x - k)$$

Ecuación 7.4.2. Fórmula del cálculo de la DWT (Torres-García, Reyes-García, Villaseñor-Pineda, & Ramírez-Cortés, 2013).

La explicación del conjunto de funciones $\psi_{j,k}(n)$ es compleja y se menciona superficialmente “referido como la familia de wavelets derivadas de $\psi(n)$, el cual es una función de tiempo con energía finita y rápido decaimiento llamada la wavelet madre. Las bases del espacio wavelet corresponden entonces, a las funciones ortonormales obtenidas de la wavelet madre después de las operaciones de escala y traslación. La definición indica la proyección de la señal de entrada en el espacio wavelet a través del producto interior” (Torres-García, Reyes-García, Villaseñor-

Pineda, & Ramírez-Cortés, 2013). No se profundiza tanto en la DWT debido a que en la presente investigación no se hace uso de este recurso.

Saltando entonces, al siguiente paso de este acercamiento al habla imaginada, se menciona que en la investigación analizada se utilizaron técnicas de aprendizaje automático para discriminar ondas cerebrales dado que “la clasificación cubre cualquier contexto en el que alguna decisión o pronóstico es hecho sobre la base de información histórica disponible” (Torres-García, Reyes-García, Villaseñor-Pineda, & Ramírez-Cortés, 2013).

La base de información disponible D se define de la siguiente forma:

$$D = \{\langle x_1, y_1 \rangle, \langle x_2, y_2 \rangle, \dots, \langle x_m, y_m \rangle\} = \langle X, Y \rangle$$

Ecuación 7.4.3. Definición de la base de información disponible D (Torres-García, Reyes-García, Villaseñor-Pineda, & Ramírez-Cortés, 2013).

Donde los valores $X_i \in X$ son típicamente vectores multi-dimensionales que poseen la forma matemática $X_i = \{z_1, z_2, \dots, z_n\}$ cuyos elementos pueden tomar valores reales o discretos. Los autores denominan a estos componentes como atributos o características y el propósito revelado en esta sección de la investigación de Torres-García y otros (2013) es inferir una función (o relación) f tal que $X \rightarrow Y$ de modo que los valores de Y estén contenidos en un conjunto finito de clases $C = \{C_1, \dots, C_k\}$ que caracterizan los datos dados.

Las técnicas de clasificación que se usan en dicha investigación son también naive Bayes, random forest y SVM, aunque no con el objetivo de compararlas entre sí, como es el caso de la presente investigación, sino como refuerzo buscando una misma conclusión tratando de demostrar que los distintos canales por los que se transmite la actividad cerebral son relevantes para proceder con investigaciones de habla imaginada.

Se recalca que las conclusiones a las que se llegaron son esperanzadoras, pero muy dependientes en gran medida del tipo de sistema que se esté desarrollando, dependiendo de “los tipos de sensores cerebrales usados (internos o superficiales), las condiciones en las que se capturan los EEG y la parte del cerebro que se pretenda estudiar. Incluso entre individuos, se experimentan variaciones de hasta un 20% en la precisión del sistema a la hora de clasificar los datos” (Marín, Martínez, Ureña, & López, 2017).

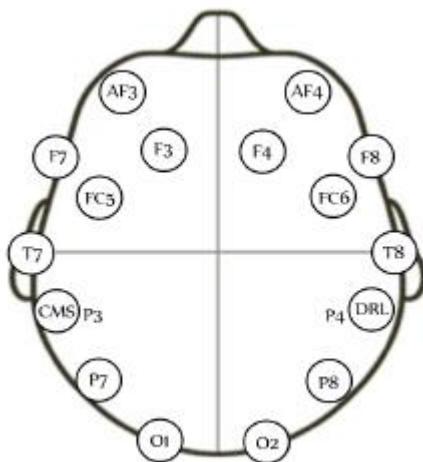


Figura 7.4.1. Ubicación de los nodos en el hardware Emotiv usado en su experimentación (Torres-García, Reyes-García, Villaseñor-Pineda, & Ramírez-Cortés, 2013).

Ellos mencionan basándose en un diagrama proporcionado dentro de su misma publicación académica – adjunto en este documento como la figura 7.4.1 – los nodos de mayor importancia a los cuales llegaron a las conclusiones mencionadas. Entre los resultados publicados, se destaca que “utilizando habla imaginada registrada únicamente de los canales F7-FC5-T7-P7, en todos los individuos, son superiores al 20%, es decir, están arriba del azar para cinco clases” (Torres-García, Reyes-García, Villaseñor-Pineda, & Ramírez-Cortés, 2013).

Remarcan que esto es curioso por haber usado palabras semánticamente similares, realizando como conjectura que las señales recopiladas podrían contener información para ser utilizada en tareas de clasificación de palabras de un vocabulario reducido. “Actualmente se está realizando un análisis de las características utilizadas por el método con la intención de ampliar nuestro conocimiento sobre cuáles de ellas son las más apropiadas para la correcta clasificación del habla imaginada” (Torres-García, Reyes-García, Villaseñor-Pineda, & Ramírez-Cortés, 2013).

Claramente también recalcan que “el usar todos los canales no necesariamente implica mayor información para obtener el mejor desempeño posible; por lo que, queda por explorar cuál es la mejor combinación de canales” (Torres-García, Reyes-García, Villaseñor-Pineda, & Ramírez-Cortés, 2013) dentro de todo el espacio de combinaciones que ellos exponen dentro de su artículo para optimizar la interpretación de las señales obtenidas del cerebro.

La experimentación en la presente investigación será similar aunque no igual a la presentada en el artículo de Torres-García y otros (2013) dando por hecho que la información del EEG sí rescata elementos con los que podríamos considerar realizar interpretaciones que lleven a decodificar esta información y usarla como medio para posteriormente desarrollar algoritmos que permitan decodificar estos datos y convertirlos en las interpretaciones del habla imaginada.

7.4.2. Breviario: evocación de un concepto

Otro proceso mental que cabe mencionar, pese a ser el de menor importancia en esta investigación es la evocación de un concepto. Este proceso mental, según Anderson (1993) se define como un proceso mental en el que la mente de una persona se activa a través de la representación mental de un objeto, idea o categoría abstracta.

Este proceso involucra diversas áreas del cerebro, según Barsalou (1999) principalmente la corteza prefrontal, la corteza temporal y la corteza parietal. Estas áreas trabajan juntas para reconocer y recordar conceptos y para relacionarlos con otros conceptos almacenados en memoria.

La evocación de un concepto puede tener muchos parecidos con el habla imaginada, sobre todo por su naturaleza de trabajo cerebral que puede estudiarse desde las BCI; involucrando distintas secciones cerebrales pero otorgando como resultado uno similar – mas no el mismo – es por ello que puede tenerse en cuenta como un proceso mental de importancia.

Se menciona en la presente investigación no por descubrimientos previos como sí se menciona el habla imaginada, sino por las instrucciones que se le piden a los usuarios que recaen como este proceso mental que se verán posteriormente, pues, este no era el núcleo de la investigación y quería trabajarse bajo la comparación de este proceso mental y la intención de movimiento pero enfatizando mucho más el segundo proceso, pues a partir de este y de las órdenes que se le dan a los usuarios que terminan por reincidir en esta área de procesos mentales, es que puede dársele el enfoque a esta investigación como una orientación al control de un equipo de cómputo.

Es así, que se abordará el último concepto, el proceso mental de esta investigación, la intención de movimiento con el que se cierra el marco teórico de esta exploración y con base en estos dos procesos mentales – pero mayormente con el próximo – se impulsará la parte experimental.

7.4.3. El proceso mental de esta investigación: intención de movimiento

Finalmente se profundizará en el proceso mental principal de esta investigación, referente a la intención del movimiento. Este proceso mental involucra la planificación, organización y ejecución de un movimiento motor específico. Según Fadiga, Craighero y Olivier (2005), esta planificación puede ser tanto consciente como inconsciente, y puede involucrar diferentes niveles de complejidad, desde movimientos simples como levantar un brazo hasta movimientos complejos como realizar una rutina de baile o deportiva.

La intención del movimiento está altamente relacionada con el sistema motor del cerebro y otros procesos mentales, que incluye áreas como la corteza motora, el cerebelo y los ganglios basales, tal como Barsalou (1999) lo expone. Estas áreas trabajan juntas para controlar la ejecución de

movimientos corporales, y también están involucradas en la planificación y la organización de las acciones.

La planificación de la intención del movimiento puede ser influenciada por diversos factores, como los expuestos según Fadiga, Craighero y Olivier (2005) quienes mencionan elementos como la experiencia previa, la atención y la motivación. Por ejemplo, las personas que tienen más experiencia en un deporte o actividad física tienden a tener una planificación de movimiento más precisa y eficiente, mientras que la falta de atención o la motivación pueden afectar negativamente la planificación del movimiento.

Además, la intención del movimiento también está relacionada con la percepción sensorial del cuerpo y del entorno. Barsalou (1999) expone que la información sensorial proporcionada por los sentidos, como la visión y la propiocepción, se utiliza para guiar la planificación y la ejecución de movimientos. La percepción sensorial también permite al cerebro ajustar la planificación del movimiento en función de los cambios en el entorno o en la posición del cuerpo.

Es muy importante enfatizar qué tipo de instrucción llevará a cabo el usuario. Es con este tipo de instrucciones con las que se determina qué proceso mental está haciendo el usuario y, por ende, cuál de los tres expuestos está examinándose.

Con lo anterior previamente expuesto, se han cubierto los tópicos biológicos del cerebro y la forma de acción de las neuronas, se mencionaron los dispositivos que permiten colecciónar esta información, siendo los EEG protagonistas de dicha sección, se destacó el papel de la computación, la inteligencia artificial, el aprendizaje automático y el énfasis en el aprendizaje supervisado, denotando la importancia de las técnicas utilizadas; ahora se habló del problema de investigación: la intención de movimiento.

El siguiente paso en esta investigación será acomodar todas estas piezas ahora en un esquema práctico y observar: ¿qué técnica de aprendizaje supervisado clasifica con mayor eficiencia datos provenientes de señales electroencefalográficas interpretables como parte de intenciones de movimiento de un individuo que desea operar un equipo de cómputo por este medio? Esta pregunta se responderá a nivel práctico en los próximos capítulos.

8. Material y método

La vida es aquello que va sucediendo mientras te empeñas en hacer otros planes.
John Lennon.

Las consideraciones de elementos que se toman para el desarrollo de esta investigación son, en principio, un equipo de cómputo con sistema operativo Windows 11 en el que se tenga instalado como software el lenguaje de programación Python en su versión 3.10, el software Emotiv en su versión 1809 para Windows, el programa CyKit en su versión 3.0, el programa OpenViBE en su versión 2.7.2 para sistemas de 64 bits. Otro elemento de hardware necesario para la recepción de datos fue la diadema Emotiv EPOC+ que a su vez se conectaría al equipo de cómputo.

Además, fue necesario contar con usuarios voluntarios a la prueba de la diadema que realizar trabajo mental según las instrucciones que se les iban pidiendo en concentrarse en ciertos conceptos o acciones. Este grupo de voluntarios está conformado por 27 usuarios de los cuales, 21 voluntarios son jóvenes de entre 18 y 25 años de edad y seis voluntarios son adultos de 30 años de edad o más. También cabe destacar que este grupo de 27 usuarios se conforma de 11 mujeres y 16 hombres. Los voluntarios se detallan en el anexo 7.

La metodología de esta investigación se planteó con base en comparación y contraste de resultados. Primeramente se eligieron los tres modelos de aprendizaje supervisado ya mencionados en capítulos anteriores, posteriormente se dedicó también tiempo a la implementación del enlace entre el dispositivo Emotiv y el equipo de cómputo para simular la Interfaz Cerebro – Computadora y coleccionar los datos mediante código Python desarrollado propiamente y para estos fines.

Se limpió levemente el conjunto de datos, añadiendo la etiqueta del concepto que se le pide al individuo pensar y se separará en distintos perfiles, con el argumento que los distintos individuos tendrán distintas intensidades y peculiaridades en la actividad cerebral, evitando que el modelo se ensucie con una mezcla de todos los conjuntos de datos compuestos como si fuera uno solo.

Después, debido a las carencias de realizar el experimento y la predicción en vivo mientras se transmiten los datos, la predicción se realizó posterior a la recolección de datos de modo que pudiera tenerse una etapa de preprocesamiento de datos. Esta etapa involucró pasar todos los conjuntos de datos por técnicas de normalización y transformación, eligiendo respectivamente la técnica de normalización y la transformada de Fourier como elementos que ayudarían a afinar el conjunto de datos y los que sirvieron como entrada para cada modelo de machine learning.

Los modelos fueron implementados con base en un código Python escrito para evaluar el conjunto de datos, eligiendo una porción del conjunto de datos como conjunto de entrenamiento y el resto del conjunto como conjunto de prueba. Finalmente, la eficiencia fue evaluada utilizando una matriz de confusión y así poder determinar la efectividad de cada modelo propuesto.

Posteriormente se trataron los datos mediante procedimientos de normalización y transformación para mejorar la eficiencia de predicción sobre cuál es el modelo que mejor se adapta a la interpretación de la actividad cerebral como intención de movimiento y a partir de estos resultados, se observaron los comportamientos de las muestras para determinar el modelo que mejor realiza la predicción, dando así por finalizada esta investigación y reportando los resultados en el presente documento. El procedimiento previamente relatado puede observarse a modo de diagrama en la figura 8.1 el cual marca el flujo previamente explicado.

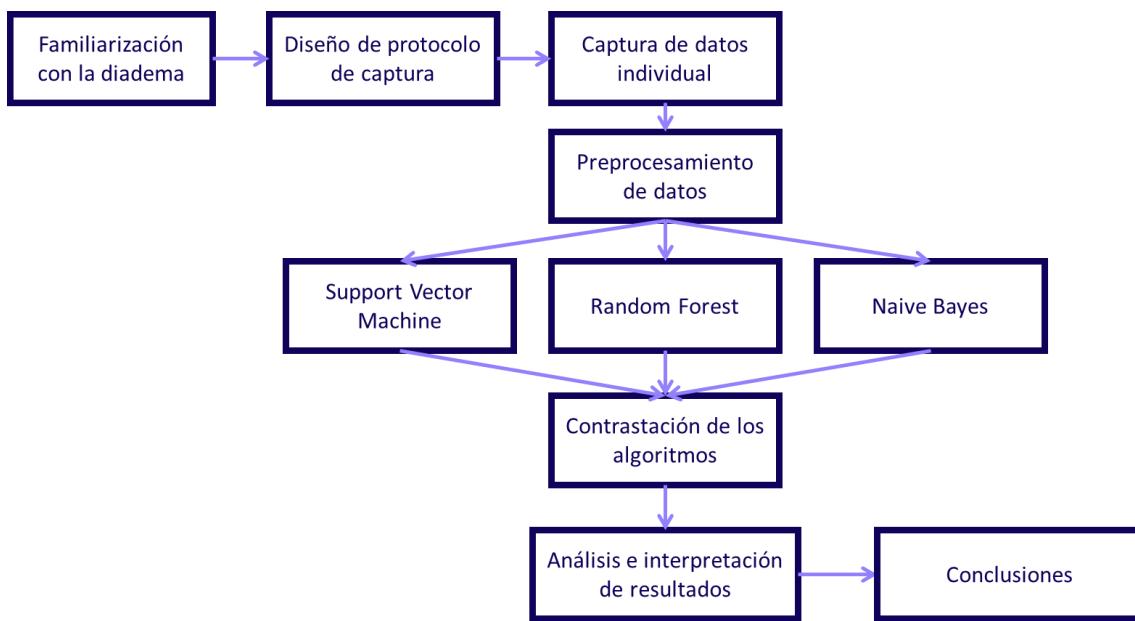


Figura 8.1. Diagrama de pasos a seguir según la metodología planteada.

9. Desarrollo

El trabajo duro no es suficiente. Tienes que creer en ello.
Derrick Rose.

Como parte del desarrollo de esta investigación primero se tuvo el periodo previo a la formalización experimental tratando de adaptarse al hardware y software requerido realizando algunas instalaciones destacables, pues para este proyecto es requerido que la instalación de la diadema y el software se haga de la manera adecuada. Para ello cabe mencionar que la presente investigación se realizó sobre el sistema operativo Windows 11, máquina a la que se le conectaría el modelo de la diadema ya mencionado. También se usará el lenguaje de programación Python en su versión 3.10, el software Emotiv en su versión 1809, el programa CyKit en su versión 3.0, el programa OpenViBE en su versión 2.7.2 para sistemas de 64 bits; instalaciones que se cubrirán en el avance de configuración de la diadema con el equipo para recibir y transmitir los datos.

La preparación de la diadema es sencillo de describir, pero tardado de realizar. La diadema se extrae de su caja y sus nodos de recepción vendrán desmontados pues las almohadillas que se usan como apoyo sobre la cabeza están separadas de los nodos y a su vez éstos de la diadema. Primeramente hay que resaltar que los nodos tienen que estar humedecidos en agua con solución salina. En el caso propio se utilizó un recipiente de 150 mililitros aproximadamente al que se le añadió cinco gramos de sal y, con una jeringa se extraía agua y sobre cada almohadilla se dejaban caer de cinco a ocho gotas por almohadilla viendo que ganara cierta humedad, como se aprecia en la figura 9.1.



Figura 9.1. Almohadillas humedecidas en solución salina.

Posteriormente, en una caja se encuentran los nodos de recepción de la diadema a los que hay que retirar de dicha caja y acomodarles una almohadilla por nodo, así como acomodar cada nodo en cualquier posición indistinta de la diadema. La figura 9.3 muestra la caja de nodos vacía y la diadema completa con todos sus nodos y almohadillas ya acomodados en su posición.



Figura 9.2. Diadema lista para su uso.

Posteriormente viene el proceso de instalación, que su descripción a detalle para tener preparada cada configuración de transmisión y preparación de datos es extenso y no es propio de los objetivos de la investigación, sino sólo es parte de los procedimientos necesarios para poder realizar correctamente la experimentación de esta investigación, por ello el proceso detallado está explicado en el anexo 1.

A grandes rasgos, es necesario tener instalado Python en una versión 3.7.0 o superior y descargar el repositorio de CyKit desde la liga <https://github.com/CymatiCorp/CyKit>. Asimismo debe de seguirse el tutorial detallado por CymatiCorp en la liga <https://github.com/CymatiCorp/CyKit/wiki/How-to-Stream-Data-to-OpenViBE>, en el cual será necesario descargar e instalar OpenViBE en una versión 2.7.2 o superior y crear un escenario como el ilustrado en la figura 9.3.

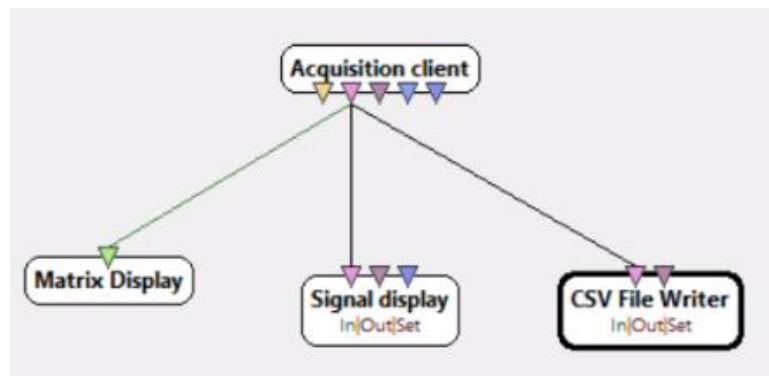


Figura 9.3. Escenario de OpenViBE Designer para experimentación de esta investigación.

Cuando se tenga preparado el escenario de OpenViBE Designer, puede conectarse la diadema y detectarla desde el software de Emotiv para posteriormente ejecutarse CyKit con las configuraciones necesarias, después ejecutar OpenViBE Acquisition Server y conectarlo a CyKit e iniciar la transmisión de datos, de modo que pueda reproducirse el escenario de OpenViBE Designer lo cual mostrará la actividad cerebral en un electroencefalograma y una matriz de datos, así como escribirlo en un archivo CSV que se haya indicado en el escenario de OpenViBE Designer.

Posteriormente se redactó el archivo de Python `functions.py` el cual se encuentra en el anexo 2 para realizar distintas acciones. A continuación se enlistan las funciones que están en el archivo y los procedimientos que realizan.

- `MousePosition` obtiene la posición del mouse pues posteriormente se usarán estas coordenadas para que el proceso de recepción de datos se realice automáticamente.
- `RunUpdateStream` es una función depreciada que buscaba realizar la transmisión de datos en vivo pero sólo se mantuvo como una función de apoyo para la siguiente (pseudocódigo presentado en el anexo 3).
- `createDataset` usa coordenadas para maximizar OpenViBE Designer, iniciar y detener el escenario para que el proceso lo realice automáticamente esta función, Así también recibe un el nombre de un perfil al que le creará una carpeta dentro del directorio de `profiles` para analizar su conjunto de datos separado al resto. También recibe un valor de resultado esperado con el que se etiquetarán estos datos como conjunto de datos de aprendizaje supervisado. Finalmente se le indica el tiempo de evaluación en segundos (pseudocódigo presentado en el anexo 4).
- `SVM` realiza el algoritmo de support vector machine sobre el conjunto de datos de un perfil determinado revisando si la carpeta de este perfil existe. Si no existe, no realiza nada.
- `NaiveBayes` realiza el algoritmo de naive Bayes sobre el conjunto de datos de un perfil determinado revisando si la carpeta de este perfil existe. Si no existe, no realiza nada.
- `RandomForest` realiza el algoritmo de random forest sobre el conjunto de datos de un perfil determinado revisando si la carpeta de este perfil existe. Si no existe, no realiza nada.
- `runModels` se creó para automatizar el proceso de evaluación (pues el tiempo de cómputo era considerable) de modo que se indica una lista de perfiles a los que se les quiere realizar el algoritmo de support vector machine, naive Bayes y random forest, registrando los resultados en un archivo `results.txt`.
- `NormZ` normaliza el conjunto de datos de un perfil indicado usando la normalización z explicada en capítulos anteriores. Estos conjuntos de datos se guardan en un directorio llamado `normz` que posee la misma estructura que `profiles`.

- Fourier transforma el conjunto de datos de un perfil indicado usando la transformada de Fourier explicada en capítulos anteriores. Estos conjuntos de datos se guardan en un directorio llamado `fourier` que posee la misma estructura que `profiles`.
- FourierEMF separa el conjunto de datos previamente transformado de un perfil indicado usando el enfoque magnitud y fase en las distintas columnas que esto proporciona, creando una columna de magnitud y otra de fase por cada canal que había previamente. Estos conjuntos de datos se guardan en un directorio llamado `fourierEMF` que posee la misma estructura que `profiles`.

De este modo, `MousePosition` se ejecuta antes de cualquier experimentación para calibrar los algoritmos siguientes, `createDataset` es la función que se ejecuta cada vez que se inicia la experimentación con algún usuario y las funciones siguientes a ésta se ejecutan después como parte del preprocesamiento de datos.

`runModels` activa las funciones `SVM`, `NaiveBayes` y `RandomForest` para evaluar la eficiencia de predicción de cada modelo de machine learning. Las funciones `NormZ`, `Fourier` y `FourierEMF` realizan el preprocesamiento de normalización, transformación y orientación a magnitud y fase respectivamente de modo que posteriormente podría ejecutarse `runModels` cambiando el directorio a `normz` para evaluar los conjuntos normalizados y cambiar el directorio a `fourierEMF` para evaluar los conjuntos de datos transformados.

Este es el desarrollo presentado de la experimentación perteneciente a esta exploración. La realización y documentación de sus pasos en la práctica se presentan en la sección siguiente.

10. Experimentación y pruebas

*Esas veces en las que estás muy cansado, no quieres exigirte pero igualmente lo haces...
Ese es el sueño. No es el destino, es el trayecto*
Kobe Bryant

Una vez planteada la estructura teórica de cómo se realizaría la experimentación presentada en el capítulo anterior, se procedió a actuar de forma práctica. La diadema se preparó de la forma narrada en el capítulo pasado en cada prueba por voluntario pero para que cada voluntario pudiera participar en la experimentación se les pidió aceptar un acuerdo de privacidad en el que se garantizaba que la información generada en la prueba sería usada de forma responsable, además de solicitar el consentimiento de realizar documentación audiovisual y utilizar el nombre y las evidencias audiovisuales generadas durante la prueba como parte de las evidencias de esta investigación (el acuerdo de privacidad está presentado en el anexo 5).

Un ejemplo de algunos participantes utilizando la diadema después de la preparación ya descrita se aprecia en la figura 10.1. Evidencias de todos los participantes en el proceso de prueba pueden encontrarse en el anexo 7.



Figura 10.1. Ejemplo de uso de la diadema en los voluntarios 1, 8, 7, 9 y 23 de izquierda a derecha, de arriba abajo.

Antes de comenzar con la prueba con cada usuario, se debía ejecutar la función MousePosition para obtener la posición del mouse en los distintos puntos de la pantalla requeridos. Un ejemplo de

la salida que proporcionaba esta función se observa en la figura 10.2 donde observamos la posición del mouse en x y en y en valores de píxeles de la pantalla.

```
In [2]: MousePosition()
Point(x=906, y=237)
```

Figura 10.2. Resultado de ejecutar la función MousePosition.

Este proceso se ejecutó para conocer la posición del mouse donde se le daría click para maximizar la ventana de OpenViBE Designer, la posición donde se acciona la reproducción del escenario de OpenViBE Designer, también donde se detiene el escenario y un punto donde pueda dársele click a OpenViBE Designer para mantenerlo como la ventana que se encuentre hasta el frente, siendo estos valores, los primeros ocho que se indican en la función createDataset.

En el caso personal, tras encontrar las distintas posiciones del mouse para las acciones requeridas, la función se ejecutaba bajo los siguientes parámetros:

```
createDataset(1211, 1048, 453, 77, 452, 15, 398, 77, 'perfil', "resultado", 10)
```

Sólo falta definir el nombre del perfil y el resultado esperado. El nombre del perfil se estableció algún nombre familiar correspondiente al voluntario terminado en un “1” o un “2” para diferenciar de las dos examinaciones que se realizaban en la experimentación.

La primera examinación giraba en torno a la evocación de un concepto, donde se les pidió a los usuarios pensar y visualizar en los siguientes conceptos:

- Árbol
- Computadora
- Cuaderno
- Perro

El proceso de esta primera examinación se tomó más que nada como calentamiento para la segunda examinación o como control comparativo frente a la segunda evaluación, en el que se les pidió a los usuarios concentrarse en intentar mover el ratón o *mouse* de una computadora en las cuatro direcciones básicas: hacia arriba, hacia abajo, hacia la izquierda y hacia la derecha volviendo esta evaluación, una tal que involucraba como proceso mental la intención de movimiento.

En ambas evaluaciones se pidió una toma de control tratando de no realizar concentración en un pensamiento en específico.

De esta forma las etiquetas que se usaban como resultado esperado para rotular cada dato de los conjuntos de datos fueron, para la primera examinación – de aquí en adelante, llamada examinación de evocación de un concepto – las siguientes:

- Arbol
- Computadora
- Cuaderno
- Perro
- Nada

Las etiquetas para la examinación de intención de movimiento fueron las siguientes:

- MouseUp
- MouseDown
- MouseLeft
- MouseRight
- Nada

Referenciando a cada intención de mover el mouse en alguna de las cuatro direcciones básicas.

Ahora, el valor del perfil, como se comenta es establecido bajo algún nombre familiar personal correspondiente al voluntario y terminado en un “1” o un “2” para diferenciar entre la examinación de evocación de un concepto o la examinación de intención de movimiento. Por ejemplo, la evaluación del participante número cinco se estableció como Roman1 para la evaluación de evocación de un concepto, mientras que se nombró como Roman2 para la evaluación de intención del movimiento. Los nombres antes de la aplicación del número pueden encontrarse en el anexo 7. Reproducir la forma de nombrar los perfiles es semejante para cada voluntario.

Con esto mencionado, finalmente se alude a la realización de la experimentación. El proceso de experimentación de campo se estandarizó mediante un protocolo el cual puede encontrarse a mayor profundidad en el anexo 6 donde se detallan los pasos que se siguieron previo a la experimentación, durante este periodo y después del mismo. La experimentación de campo se realizó bajo este protocolo con todos los voluntarios obteniendo documentación audiovisual del participante como de lo que sucedía en la pantalla del equipo de cómputo donde se realizaban las pruebas.

```

In [2]: createDataset(1211, 1048, 453, 77, 452, 15, 398, 77, "Yissel1",
"Arbol", 10)

In [3]: createDataset(1211, 1048, 453, 77, 452, 15, 398, 77, "Yissel1",
"Cuaderno", 10)

In [4]: createDataset(1211, 1048, 453, 77, 452, 15, 398, 77, "Yissel1",
"Computadora", 10)

In [5]: createDataset(1211, 1048, 453, 77, 452, 15, 398, 77, "Yissel1",
"Perro", 10)

In [6]: createDataset(1211, 1048, 453, 77, 452, 15, 398, 77, "Yissel1",
"Nada", 10)

In [7]: createDataset(1211, 1048, 453, 77, 452, 15, 398, 77, "Yissel2",
"MouseUp", 10)

```

Figura 10.3. Ejemplo de ejecución de la función createDataset con el voluntario 24.

En consola, no se mostraban resultados visibles tras ejecutar la función createDataset como se observa en la figura 10.3 ejemplificando la ejecución continua del participante 24, sin embargo, realizar las dos examinaciones por persona sí daba como resultado dos archivos separados por comas como el que se aprecia en la figura 10.4.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	Expected Output
1	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6	Channel 7	Channel 8	Channel 9	Channel 10	Channel 11	Channel 12	Channel 13	Channel 14		
2	4216.02588	4927.30762	4159.87158	4161.6665	4158.87158	4161.15381	4156.53857	4160.38477	4167.17969	4168.3335	4165.38477	4167.30762	4168.3335	4165.38477	4167.30762	
3	4154.4873	4157.94873	4158.58984	4156.41016	4163.20508	4168.46143	4167.56396	4163.07715	4173.07715	4166.21928	4160.25635	4161.6665	4158.6665	4160.25635	4168.6665	
4	4164.87158	4166.79492	4163.58984	4172.69238	4215.5127	4297.69238	4159.4873	4161.53857	4157.43604	4167.82031	4165.41016	4160.76904	4158.58984	4160.25635	4168.6665	
5	4924.74365	4160.76904	4163.07715	4156.32232	4167.43604	4159.4873	4160.25058	4158.20508	4162.43604	4168.07715	4166.52285	4165.5127	4168.79412	4215.64111	4167.69238	
6	4156.79492	4160.5127	4156.02588	4160.38477	4166.79492	4166.20508	4160.25058	4170	4214.61523	4923.97412	4156.6665	4162.05127	4156.53857	4156.53857	4168.6665	
7	4167.69238	4167.69238	4167.43604	4215.5127	4925.12842	4157.56396	4159.61523	4159.10254	4157.69238	4155.25635	4157.94873	4155.76904	4158.71777	4167.69238	4168.6665	
8	4155.89746	4161.02588	4158.20508	4167.05127	4153.58984	4156.6665	4153.58984	4156.53857	4158.71777	4167.56396	4171.15381	4168.58984	4218.58984	4921.6665	4168.58984	
9	4158.97412	4155.5127	4157.43604	4159.61523	4166.6665	4173.46143	4163.97412	4217.69238	4927.82031	4156.15381	4158.84619	4161.02588	4168.20508	4159.10254	4168.20508	
10	4174.87158	4164.61523	4215.12842	4292.61523	4156.15381	4160.38477	4156.53857	4166.92285	4155.25635	4159.87158	4160.89746	4167.56396	4167.94873	4168.6665	4167.94873	
11	4158.71777	4155	4161.28223	4155.5127	4158.3335	4159.10254	4160	4165.38477	4168.20508	4172.05127	4171.6665	4216.15381	4923.07715	4155.76904	4168.6665	
12	4160.5127	4159.25635	4169.35889	4166.92285	4170.38477	4214.10254	4925.25635	4158.71538	4160.76904	4164.87158	4155.76904	4158.71777	4167.69238	4168.6665	4168.6665	
13	4166.15381	4216.28223	4169.27969	4160.25635	4160.64111	4156.02588	4167.05127	4156.79492	4159.4873	4160.89746	4159.87412	4168.3335	4165.6665	4165.6665	4168.3335	
14	4157.56396	4158.97412	4158.3335	4158.58984	4161.6665	4158.97412	4160.89746	4168.20508	4162.43604	4167.05127	4156.15381	4158.84619	4160.38477	4159.4873	4160.76904	
15	4156.15381	4163.71777	4167.56396	4158.41024	4160.12842	4215.76904	4922.05127	4156.79492	4158.46143	4165.38477	4158.71777	4160.89746	4159.4873	4160.38477	4168.6665	
16	4215.89746	4921.79492	4155.64111	4158.58984	4159.23096	4168.58984	4158.84619	4156.79492	4160.76904	4160.89746	4161.41016	4167.30762	4167.05127	4157.56396	4168.6665	
17	4168.20508	4157.56396	4156.53857	4161.41016	4160.5127	4164.74365	4166.02588	4166.92285	4154.23096	4155.25635	4162.41016	4160.89746	4168.6665	4158.07715	4168.6665	
18	4162.56396	4166.28223	4158.41016	4215.97412	4158.46143	4215	4923.84619	4158.3335	4162.43604	4158.46143	4165.38477	4160.89746	4158.71777	4156.92285	4158.84619	
19	4923.46143	4157.94873	4161.28223	4157.69238	4166.15381	4158.46143	4156.53857	4157.69238	4167.05127	4156.15381	4158.79412	4164.87158	4159.4873	4160.76904	4168.6665	
20	4157.56396	4158.97412	4157.94873	4161.51253	4166.92285	4166.79412	4173.3335	4157.56396	4161.28223	4157.69238	4161.28223	4159.74363	4159.74363	4166.15381	4166.15381	
21	4166.41016	4177.05127	4159.23096	4214.10254	4925.25635	4159.61523	4159.23096	4158.07715	4163.20508	4158.71538	4160.76904	4163.20508	4157.56396	4157.56396	4154.74365	
22	4158.20508	4160.5127	4156.28223	4170.30762	4155.64111	4159.74363	4160.41016	4157.56396	4161.28223	4157.69238	4162.41016	4217.30762	4162.41016	4157.56396	4168.6665	
23	4160.12842	4156.79492	4159.10254	4162.17969	4168.97412	4219.10254	4166.53857	4217.05127	4926.15381	4157.05127	4158.07715	4161.53857	4157.15381	4158.3335	4160.76904	
24	4177.03127	4162.17969	4216.41016	4925	4155.38477	4156.41016	4159.61523	4162.43604	4157.69238	4160	4157.43604	4158.3335	4166.15381	4166.15381	4168.6665	
25	4157.82031	4159.23096	4161.6665	4157.82031	4157.8143	4157.8143	4160.5127	4157.69487	4170.76904	4161.62323	4157.69487	4162.41016	4157.5127	4162.41016	4168.6665	
26	4154.35889	4160.5127	4165	4168.84619	4172.05128	4162.43604	4216.28223	4923.46143	4162.17969	4160.76904	4162.82031	4153.46143	4157.56396	4161.15381	4168.6665	
27	4158.58984	4218.58984	4925.12842	4162.05127	4163.46143	4160	4153.46143	4161.15381	4160	4157.94873	4159.4873	4167.43604	4172.30762	4162.41016	4168.6665	
28	418.84619	4148.97412	4159.61523	4160.25635	4160.38477	4162.30762	4168.07715	4168.6665	4161.15381	4157.94873	4162.41016	4155.76904	4162.56396	4168.6665	4168.6665	
29	4161.41016	4168.58984	4167.69238	4169.35889	4163.46143	4218.07715	4922.69238	4161.15381	4157.94873	4149.61523	4157.69238	4159.4873	4158.46143	4160.38477	4168.6665	
30	4216.41016	4928.07715	4156.02588	4160.25635	4162.30762	4155.64111	4156.53857	4159.87158	4155.25635	4160.5127	4166.02588	4167.82031	4179.10254	4156.41016	4168.6665	
31	4152.30762	4155.12842	4160.64111	4157.17969	4162.05127	4167.82031	4174.74365	4158.58984	4162.6665	4157.43604	4164.87158	4161.02588	4163.79412	4157.94873	4168.6665	
32	4163.71777	4167.69238	4159.87158	4174.61523	4162.43604	4215.76904	4923.07715	4159.74365	4159.87158	4162.43604	4156.6665	4158.46143	4159.4873	4160.25635	4168.6665	
33	4925.5127	4157.69238	4159.87158	4158.20508	4153.58984	4159.23096	4158.3335	4157.94873	4159.35889	4162.17969	4168.02508	4170.12842	4163.84619	4216.79492	4168.6665	
34	4159.4873	4158.07715	4158.84619	4159.35889	4158.84619	4167.17969	4168.46143	4162.6665	4215.25635	4163.84619	4157.94873	4162.41016	4151.28223	4168.6665	4168.6665	

Figura 10.4. Fragmento del segundo archivo obtenido de la experimentación de campo del voluntario 14.

Ya con este conjunto de datos podía evaluarse su eficiencia introduciendo estos archivos a los modelos de machine learning implementados en Python, ya sea invocando a la función únicamente del modelo de machine learning indicándole a qué perfil evaluará – como se aprecia en la figura 10.5 con el voluntario 17 precisando la matriz de confusión de la evaluación y la efectividad – o usando la función runModels e indicándole en una lista los perfiles a los que evaluaría.

```
In [4]: NaiveBayes('Mezal')
Matriz de Confusión
[[583  4   9   28  267]
 [ 50  34   1   4  35]
 [135  13   3   9  72]
 [ 35   0   0   5  60]
 [ 48   0   1   5  97]]
Precisión del modelo
[0.68507638 0.66666667 0.21428571 0.09803922 0.1826742 ]
C:\Users\alexe\Anaconda3\envs\ici-thesis\lib\site-
packages\sklearn\metrics\_classification.py:1375: UserWarning: Note that
pos_label (set to 'a') is ignored when average != 'binary' (got None). You
may use labels=[pos_label] to specify a single positive class.
  UserWarning,
Out[4]: array([0.68507638, 0.66666667, 0.21428571, 0.09803922, 0.1826742
])
```

Figura 10.5. Ejecución del algoritmo naive Bayes con el voluntario 17 en su prueba de evocación de un concepto.

Dado que los valores en primeras evaluaciones resultaron bajos, además se realizó un proceso de normalización eligiendo la Normalización Z, buscando un mejor desempeño de los algoritmos tras este tratamiento de los datos ejecutando la función NormZ. En la ejecución de esta función se debe pasar una lista de los perfiles a los que se normalizará el conjunto de datos, se recomienda declarar todos los perfiles pues no toma mucho tiempo de evaluación. La ejecución de esta función se observa en la figura 10.6.

```
In [6]: NormZ(['joul1', 'joul2', 'pep1', 'pep2'])
```

Figura 10.6. Ejecución de la función de normalización z sobre ambas evaluaciones de los participantes 1 y 2.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	Expected Output
1	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6	Channel 7	Channel 8	Channel 9	Channel 10	Channel 11	Channel 12	Channel 13	Channel 14		
2	-0.01337529	3.60784798	-0.28292767	-0.30589747	-0.30372403	-0.32002973	-0.31849711	-0.31571988	-0.31159482	-0.30061607	-0.28783556	-0.27020859	-0.28319672	-0.25520886	MouseUp	
3	-0.03480198	-0.31621783	-0.32575541	-0.30978692	-0.29074416	-0.28238865	-0.27896412	-0.29303934	-0.25125121	-0.00667662	0.35999426	-0.28967581	-0.31041155	-0.29479331	MouseUp	
4	-0.28533003	-0.28186473	-0.30563781	-0.25353444	-0.00133421	3.60754965	-0.28933363	-0.31312853	-0.29472588	-0.318135	-0.32477527	-0.32277032	-0.33438616	-0.30906813	MouseUp	
5	3.59993401	-0.29612306	-0.29460774	0.29812352	-0.32643385	-0.3187321	-0.31914629	-0.32068778	-0.31354217	-0.27985234	-0.27487235	-0.29292075	-0.25792651	-0.01576462	MouseUp	
6	-0.31648465	-0.32140198	0.23964794	-0.3020105	-0.29074416	-0.27589804	-0.30034985	-0.26575046	-0.01181916	3.59714974	-0.28977806	-0.30135564	-0.30587574	-0.32074797	MouseUp	
7	-0.27105175	-0.29353099	-0.25502254	-0.00652865	3.595544	-0.28758163	-0.31525618	-0.29239026	-0.31768695	-0.32072985	-0.31505286	-0.31563292	-0.30133994	-0.29219587	MouseUp	
8	-0.30415432	-0.29806836	-0.29785275	-0.31820993	-0.32124388	-0.317432	-0.32044217	-0.28332051	-0.27331228	-0.27011795	-0.30533293	-0.2617737	-0.02020726	3.6141885	MouseUp	
9	-0.31908273	-0.3285339	-0.29590525	-0.29034741	-0.27322431	-0.3187321	-0.25757592	-0.0137527	3.59783384	-0.28850713	-0.29885132	-0.30589807	-0.30069091	-0.32334542	MouseUp	
10	-0.29311931	-0.26112262	-0.00454139	3.57878657	-0.27906426	-0.30501331	-0.28479681	-0.29368841	-0.32327652	-0.32462459	-0.31440618	-0.31044051	-0.2907274	-0.2662412	MouseUp	
11	-0.29571491	-0.29418024	0.30882338	-0.3182099	-0.3194639	-0.32262499	-0.31823857	-0.28220237	-0.28434403	-0.27531012	-0.24635675	-0.31944023	-0.3093523	-0.28635599	MouseUp	
12	-0.32622187	-0.2840154	-0.28033018	-0.2845158	-0.2985159	-0.2492905	-0.01713207	3.59753842	-0.28888624	-0.27290328	-0.3221034	-0.32920133	-0.3239292	-0.3239292	MouseUp	
13	-0.2496318	-0.00962947	3.60601345	-0.29228935	-0.29334162	-0.28628153	-0.33405261	-0.3209026	-0.32003174	-0.31359278	-0.30181429	-0.2961657	-0.27153393	-0.30842062	MouseUp	
14	-0.28273443	-0.32010594	-0.31212784	-0.31820993	-0.3303288	-0.29212209	-0.28220258	-0.264527	-0.30251287	-0.25454639	-0.00721865	3.58556091	-0.28967714	-0.30517567	MouseUp	
15	-0.28857577	-0.2915857	-0.26800259	-0.28905144	-0.26024692	-0.00657487	3.58435197	-0.29109458	-0.30835249	-0.29737127	-0.32931119	-0.33380265	-0.31559391	-0.32788785	MouseUp	
16	-0.01337529	3.58645964	-0.29720726	-0.30136143	-0.30939813	-0.32327504	-0.31050357	-0.33170983	-0.29282805	-0.30205094	-0.31044155	-0.26299625				
17	-0.18143426	-0.29871761	-0.31861786	-0.33181806	-0.32024654	-0.29342219	-0.28155588	-0.3016686	-0.26228298	-0.02030328	3.60707755	-0.2909732	-0.30069091	-0.29333575	MouseUp	
18	-0.30025843	-0.27732681	-0.23995775	-0.2560035	-0.00717416	3.60560445	-0.28686445	-0.28785662	-0.28785662	-0.31164541	-0.31638469	-0.33244154	-0.29998325	-0.28563062	MouseUp	
19	3.578516	-0.29871761	-0.28162769	-0.29941918	-0.30437154	-0.312451	-0.30747842	-0.31831124	-0.29407593	-0.28049982	-0.2755251	-0.28967581	-0.25663092	-0.00732972	MouseUp	
20	-0.31129873	-0.32334727	-0.32121889	-0.29877012	-0.2756480	-0.32086062	-0.30099902	-0.28220258	-0.27330227	-0.32090271	-0.3026556	-0.30004435	-0.32788785	MouseUp		
21	-0.26910381	-0.30066043	-0.24399246	-0.0181920	3.60008646	-0.29147451	-0.30488666	-0.30146248	-0.31613701	-0.31878495	-0.32283029	-0.33964257	-0.30004435	-0.28960089	MouseUp	
22	-0.30090858	-0.30195647	-0.29590525	-0.32015466	-0.31799892	-0.322976317	-0.33729108	-0.29174366	-0.27850443	-0.28244719	-0.28977806	-0.26631613	0.00384718	3.60445612	MouseUp	
23	-0.31908273	-0.32878484	-0.23995775	-0.28062595	-0.28424527	-0.29082446	-0.25239239	-0.1051227	3.59588985	-0.28309713	-0.29885132	-0.31300273	-0.318153	MouseUp		
24	-0.2729969	-0.23262027	-0.12328272	3.5910990	-0.29398813	-0.301858	-0.29192539	-0.31507081	-0.31743443	-0.30840308	-0.32347944	-0.29162327	-0.2961551	-0.27597358	MouseUp	
25	-0.2905237	-0.30778988	-0.31018034	-0.3108258	-0.33162629	-0.33820146	-0.29775809	-0.29329026	-0.26617529	-0.2907385	-0.25413418	-0.00610737	3.57925834	-0.29098938	MouseUp	
26	-0.34050015	-0.30260572	-0.28552269	-0.26507558	-0.28230923	-0.25448348	-0.0048175	3.5839301	-0.28561417	-0.30450834	-0.29431469	-0.30784554	-0.3227209	-0.31945049	MouseUp	
27	-0.26391261	-0.0070374	3.59433339	-0.28127502	-0.30696652	-0.3057508	-0.30877677	-0.32155167	-0.32587136	-0.33630388	-0.29366801	-0.2890288	-0.2592221	-0.28246349	MouseUp	
28	-0.31259126	-0.31232725	-0.2316039	-0.32598636	-0.35044363	-0.29926275	-0.28738857	-0.27952651	-0.28693889	-0.26038604	0.00185461	3.59659077	-0.29880413	-0.28505846	MouseUp	
29	-0.29246917	-0.29093891	-0.27708764	-0.29295842	-0.25446097	0.0012109	3.59018466	-0.29433502	-0.31091349	-0.31051623	-0.32536529	-0.31624294	-0.32139795			
30	-0.00299041	3.59488153	-0.28292767	-0.30589747	-0.30566903	-0.3193968	-0.32751056	-0.33403058	-0.29477642	-0.28848224	-0.27994343	-0.30846993	-0.26299625	MouseUp		
31	-0.321028	-0.32529257	-0.29847494	-0.3395944	-0.3012657	-0.28628153	-0.28609269	-0.29887113	-0.2694200	-0.02095322	3.57985777	-0.29551572	-0.3078179	-0.30063324	MouseUp	
32	-0.29246917	-0.26825207	-0.2887677	-0.25729916	-0.00133421	3.58029453	-0.30794334	-0.2979706	-0.32981429	-0.30922041	-0.3136854	-0.33049692	-0.29089838	MouseUp		
33	3.59604061	-0.28899362	-0.30239527	-0.28969803	-0.308914	-0.31353912	-0.33210756	-0.35071061	-0.28930466	-0.29241647	-0.28977806	-0.29551572	-0.26460464	-0.01835599	MouseUp	
34	-0.322326	-0.32658886	-0.32770291	-0.29358748	-0.29658412	-0.28693158	-0.30034985	-0.25804861	-0.02739076	3.57963081	-0.29885132	-0.30200562	-0.30458015	-0.32723787	MouseUp	

Figura 10.7. Fragmento del segundo archivo obtenido tras la normalización z del voluntario 26.

Un ejemplo de cómo resultan los conjuntos de datos tras la ejecución de la función de normalización puede visualizarse en la figura 10.7.

Esto no fue suficiente para tener resultados satisfactorios, por lo que posteriormente se les aplicó a los datos una transformada de Fourier bajo el enfoque de magnitud y fase que puede verse la ejecución de los comandos en las figuras 10.8 y 10.9; lo cual resultó en dos columnas nuevas por cada canal previo, es decir, un conjunto de datos como el que se visualiza en la figura 10.10.

```
In [6]: Fourier(['Abraham1', 'Abraham2', 'Valeria1', 'Valeria2', 'Juliett1', 'Juliett2'])
```

Figura 10.8. Ejecución de la función de transformada de Fourier sobre ambas evaluaciones de los participantes 12, 13 y 14.

```
In [6]: FourierEMF(['Abraham1', 'Abraham2', 'Valeria1', 'Valeria2', 'Juliett1', 'Juliett2'])
```

Figura 10.9. Ejecución de la división en enfoque magnitud y fase sobre ambas evaluaciones de los participantes 12, 13 y 14.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	Expected Ou	Channel 1 M	Channel 1 F	Channel 2 M	Channel 2 F	Channel 3 M	Channel 3 F	Channel 4 M	Channel 4 F	Channel 5 M	Channel 5 F	Channel 6 M	Channel 6 F	Channel 7 M	Channel 7 F	Channel 8 M	Channel 8 F	Channel 9 M	Channel 9 F	Channel 10 AChar
2	MouseUp	1.81e-11	3.14089232	2.69e-12	3.13977537	2.90e-12	0.00061284	2.19e-11	-3.14148136	7.40e-13	-0.01020373	1.05e-11	3.13922088	1.18e-11	-0.0126726	3.57e-12	-0.0311379	4.12e-13	-0.06692692	1.06e-12
3	MouseUp	2.93990059	1.25081212	-0.89289976	5.84901209	3.10237807	1.87269901	-3.12849875	2.77383383	-0.88893732	4.48538247	0.0126739	5.4633156	1.81409506	8.05810508	-2.63468713	1.96462983	2.9		
4	MouseUp	3.46682793	1.87574368	3.52779372	1.82553404	3.10685689	-0.50067116	0.46768427	-2.15047776	0.50591637	-1.20820216	3.73321054	2.37462762	-2.39955231	3.54829881	-0.65015366	3.05627842	-1.66962159	0.51651136	-1.1
5	MouseUp	1.96459182	-1.59550162	2.66633963	-1.51722212	4.2384034	0.56872161	1.20452659	0.59333102	1.24852047	0.57774349	2.78762264	-1.4988595	1.17080447	2.51934532	0.69700563	2.77702542	1.26689267	1.42785151	0.4
6	MouseUp	3.13427357	0.61942905	2.01488349	0.35952594	7.00078594	-1.07951444	2.24876534	-2.51986775	1.15057737	-2.62164453	3.60956239	0.79427578	5.37974038	-0.27636384	6.45987347	-1.61310323	7.53073932	2.73845967	1.6922576
7	MouseUp	0.87711377	0.29093551	0.5983282	3.0499756	3.2850796	2.98260382	3.99052638	-0.7918303	3.49633339	-0.30967833	0.64758937	0.40853287	3.93977053	-1.10972873	3.96610489	3.05999339	3.05898656	-2.88677797	4.15672626
8	MouseUp	2.76186739	-0.80731314	2.0099461	-1.18326051	6.83707633	1.69160406	2.56192493	2.92641689	1.46842647	-2.8372068	2.34242068	-0.63488342	4.01699233	-0.0050231	6.17922348	1.70666672	7.9202969	6.71811951	1.28959345
9	MouseUp	3.77950678	1.70265272	3.16537122	1.69416268	3.31515006	-0.67191598	1.2617478	-0.81590387	1.32245549	-1.28580126	2.01365178	2.82302298	3.11759581	-1.89041386	9.24838965	-0.06807062	2.92598201	2.22912775	2.681301074
10	MouseUp	3.35046472	-1.45286632	3.7354342	1.6384699	3.0984606	0.34912297	0.6911614	0.38991181	1.02496114	2.11636042	3.11759581	-1.89041386	9.24838965	-0.06807062	2.92598201	2.22912775	2.681301074	1.4	
11	MouseUp	2.0721352	0.95503394	1.25796013	0.56594741	6.71945722	-1.39717024	2.30540823	0.53817909	2.13804472	2.67145959	2.93480079	-1.05972873	3.96610489	3.05898656	-2.88677797	4.15672626	1.68641381	0.94626277	
12	MouseUp	0.39050859	-0.11763364	0.60955201	0.8533523	2.3585293	2.96152915	4.60035354	0.04875684	4.0512961	-0.22309416	1.03184537	3.74003317	-2.90112019	3.2679566	0.6955639	1.65241785	3.14		
13	MouseUp	2.69140918	-0.70780787	2.54249327	-0.7262743	6.16377322	1.41466797	1.54310277	3.04089417	0.53461582	2.53538252	3.19764224	0.40398162	2.51934532	0.69700563	2.77702542	1.26689267	1.42785151	0.4	
14	MouseUp	4.45214441	1.73746033	3.0506638	1.53794048	3.51495622	-0.4432682	1.0436906	0.79363242	0.3939488	4.16473118	1.88318912	1.16159091	3.3795117	0.68529821	1.33898321	-1.68589321	1.70554097	-0.8	
15	MouseUp	2.66051017	-1.94205061	3.15085441	1.39468227	3.92518269	0.89875545	1.06488597	-0.6844177	2.05461849	2.19225046	2.03238991	1.67488862	2.19225046	1.42770549	1.3				
16	MouseUp	3.53556351	0.82469238	2.32773113	0.48991849	5.758804	1.2410657	2.98251907	2.1646800	2.68276105	1.95225678	0.48082869	4.1626129	-0.8773299	5.54886048	-1.67761429	7.07358137	2.57568828	2.45057749	
17	MouseUp	0.58483432	0.88021864	0.85638426	0.1620052	2.86027472	-2.8865632	5.31362393	0.13361764	0.69103049	2.87632211	2.87632211	0.28763220	3.87632211	4.76152625	0.0				
18	MouseUp	3.10079242	-0.80907454	2.8010742	-0.48302472	6.50678783	1.3912612	2.44408041	2.12820502	-0.16690594	0.81642666	3.12282507	3.99130331	1.50784115	6.26688101	-2.70340259	0.85852321	-1.9		
19	MouseUp	3.97607088	1.76671015	4.55073519	1.66890437	3.9040617	-0.48858747	1.28250252	-1.28520502	2.05461849	2.342023889	3.0781091	0.57781193	3.23281929	-1.53261152	0.56132865	-0.2			
20	MouseUp	3.29372884	0.53465355	2.6734002	3.31498047	0.99932615	0.81274039	0.53370053	0.75826035	-0.03239499	2.42023886	-1.64199933	3.27049881	2.13434425	1.67930408	0.64382661	0.0			
21	MouseUp	3.1735108	0.78807506	2.82953824	0.86058122	6.1340406	-1.5047283	0.08822101	1.35932094	-2.61174782	3.00933384	0.84553677	5.67104673	0.0230848	1.55481471	6.88277068	2.59031915	3.05283086	2.9	
22	MouseUp	0.54340184	0.29504224	1.18632181	-0.6518258	3.35740154	-3.3174513	4.16779357	-0.31156622	4.56055727	-0.00135421	3.13963444	-2.81579311	3.77848706	3.01582746	3.71510103	-2.96058259	3.95487529	-0.0	
23	MouseUp	3.16744051	-0.87616133	2.84412901	-0.78437583	6.206799	1.51909637	1.1848238	-2.97154499	0.7646381	3.61027521	-0.92418578	1.67461622	4.32770549	1.45522733	7.34835485	-2.9			
24	MouseUp	3.80772828	1.82373001	3.80932369	1.9829701	3.11020621	-0.54947236	1.7081579	2.33102532	0.00603166	2.45744768	2.83729728	2.99302532	3.09406583	-0.6725251	3.13832165	-1.53403337	0.74487953	1.1	
25	MouseUp	3.8538529	-1.79735014	2.5054699	-1.81420269	2.83029478	0.7811793	1.09310606	0.29587581	1.08934196	0.50944851	2.66502593	-0.20123194	2.1752698	2.86709206	0.83725163	2.84711249	1.34481828	1.39590236	
26	MouseUp	2.65122518	1.70277709	2.5054699	1.58657427	-1.39086173	3.04291763	1.63224484	3.04291763	0.77514139	3.84954025	5.13695258	1.50754109	5.7575569	2.60722587	0.14394773	-2.6			
27	MouseUp	0.52816857	1.49036534	0.32517368	0.56653687	3.35465019	-2.90407878	4.719522	-0.13159594	4.6257422	-0.83954881	3.44334178	-0.05131196	4.719522	-0.86872133	3.0740929	-2.84748709	5.51499348	-0.0	
28	MouseUp	3.20117785	-1.10692419	3.03898113	-0.31425663	2.50620922	2.53020518	1.45638442	-1.68324035	2.97687236	5.93865493	0.38338604	6.81355661	1.57551059	7.8024988	-2.75495301	0.96255434	-2.4		
29	MouseUp	3.94767318	1.79172677	2.9888238	1.78608015	3.4959792	0.93522871	-1.35360063	0.8557172	-0.66683401	2.32403022	2.12817577	3.22178267	-2.73483344	3.51683822	1.69767329	0.48359918	-0.4		
30	MouseUp	4.42308936	-1.79163145	3.15763211	-1.67717603	2.33685493	0.63973503	0.44664078	0.44664078	2.69402881	2.69402881	2.1921621	0.77552628	3.66008893	1.70682778	0.6645355	1.8			
31	MouseUp	2.16621503	0.96000099	1.19172455	0.72621016	6.1800406	2.48476409	1.77181689	-2.79526029	2.34892487	0.79551123	4.71264729	0.05740425	5.44966799	1.3118822	7.18946729	2.74446309	3.00646683	-2.5	
32	MouseUp	1.18658055	-0.11815791	0.66655654	0.6600000	3.42923105	4.57700073	-0.2850892	4.30724114	0.07012604	1.49061981	-1.16250715	3.20062123	0.32203479	3.70222547	2.96759328	4.2915266	0.18		
33	MouseUp	2.71039531	-0.85469883	2.88662761	-0.43593944	7.08827856	1.82924731	2.11504062	2.48472851	1.62925097	3.05828601	5.23915942	0.26517378	5.54434783	1.44813933	7.50134721	-2.67463731	1.56674901	-2.7	
34	MouseUp	2.91675434	1.70485405	2.7433901	2.04781586	3.51817143	-1.07915251	0.94822791	0.06651859	0.97085084	3.83739247	2.0503924	2.77089922	-2.73938277	4.31146925	-1.00279592	3.49382732	-1.91696209	0.1914932	1.61

Figura 10.10. Fragmento del segundo archivo obtenido tras la transformación de Fourier del voluntario 19.

Por último se ejecutaron las funciones de los tres modelos de aprendizaje supervisado únicamente indicando el perfil que se evaluará. Con este proceso, cada función arrojará un porcentaje de efectividad después de haber separado a la población en un conjunto de entrenamiento y un conjunto de prueba, produciendo una matriz de confusión en la que será visible la efectividad del modelo. Al ejecutar estos modelos en Python, el resultado similar al que se visualiza en la figura 10.11.

```
In [4]: RandomForest('Andy2')
Matriz de Confusión
[[139  6  4 11  1]
 [ 8 183  7  2  1]
 [12 13 145  1 10]
 [19  1  2 71 16]
 [ 7  2 11 17 157]]
Precisión del modelo
[0.75135135 0.89268293 0.85798817 0.69607843 0.84864865]
C:\Users\alexe\Anaconda3\envs\ici-thesis\lib\site-
packages\sklearn\metrics\_classification.py:1375: UserWarning: Note that
pos_label (set to 'a') is ignored when average != 'binary' (got None). You
may use labels=[pos_label] to specify a single positive class.
UserWarning,
Out[4]: array([0.75135135, 0.89268293, 0.85798817, 0.69607843,
0.84864865])
```

Figura 10.11. Resultado de ejecución del modelo de random forest sobre la segunda evaluación del voluntario 10.

Finalmente este procedimiento se repitió varias veces, concretamente por modelo para cada evaluación de cada voluntario se ejecutó cinco ocasiones distintas, de modo que se obtuviera un promedio de estas ejecuciones. Todos estos resultados se almacenaron en una hoja de cálculo donde se realizaría la interpretación de resultados. Un fragmento de la hoja de cálculo puede verse en la figura 10.12 donde las ejecuciones están en celdas cuyo color es grisáceo mientras que los promedios de dichas ejecuciones están coloreados con distintas señalizaciones; naranja para SVM, azul para naive Bayes y amarillo para random forest. Los resultados son presentados a continuación.

		SVM					Naive Bayes					Random Forest						
		Arbol	Cuaderno	Computadora	Perro	Nada	Arbol	Cuaderno	Computadora	Perro	Nada	Arbol	Cuaderno	Computadora	Perro	Nada		
Joel Alejandro Espinoza Sánchez	Ronda 1	0.28968254	0.34185304	0.38461538	0.31666667	0.30769231	0.2395082	0.21008403	0.0512821	0.27072027	0.72164948	0.66666667	0.55776892	0.67530337	0.65730337			
		0.28384279	0.30116059	0.31531532	0.33333333	0.30681818	0.315185185	0.21052632	0.17913094	0.03571429	0.26923077	0.55072464	0.7534246	0.71599090	0.69186047	0.64705082		
		0.33057851	0.31498471	0.264	0.30493274	0.33333333	0.27686883	0.2511497	0.1311475	0.2551497	0.735717576	0.7543880	0.0975961	0.80952381	0.74403175	0.7679558		
José Luis Espinoza Sánchez	Ronda 2	0.26859508	0.26842189	0.21052632	0.30392157	0.36998630	0.26315789	0.23289316	0.25757576	0.17543880	0.26315789	0.59809305	0.65515837	0.75534068	0.65934068	0.72580976		
		0.28968254	0.34185304	0.38461538	0.31666667	0.30769231	0.2395082	0.21008403	0.0512821	0.27072027	0.72164948	0.66666667	0.55776892	0.67530337	0.65730337	0.64705082		
		0.29352342	0.311372634	0.309616578	0.320670384	0.32818785	0.280196849	0.210047426	0.210047426	0.179093324	0.263734696	0.5757219	0.675737698	0.715911322	0.679009293	0.649902006		
José Luis Espinoza Sánchez	Ronda 3	0.31952663	0.66666667	0.71438571	0.5	0.44827586	0.26666667	0.3943662	0.22244489	0.16666667	0.2777778	0.4253521	0.41044776	0.42498718	0.4597033	0.4280992		
		0.30689393	0.66666667	0.71438571	0.5	0.44827586	0.26666667	0.3943662	0.22244489	0.16666667	0.2777778	0.4253521	0.41044776	0.42498718	0.4597033	0.4280992		
		0.31692913	0	0.5	0.25	0.2775	0.4	0.25389232	0.09847328	0.44036697	0.27482751	0.43979085	0.46468992	0.51333333	0.42236025	0.52083333		
José Luis Espinoza Sánchez	Ronda 4	0.32443532	0.32812125	0	0.42105263	0.31333333	0.34567901	0.24074074	0.18656716	0.25146199	0.4548105	0.44936709	0.41843972	0.55479452	0.52083333	0.52083333		
		0.290003559	0.38888889	0.66666667	0.46153846	0.27672956	0.36363636	0.23766816	0.16666667	0.40322581	0.28481018	0.42105263	0.55714282	0.33027523	0.48780488	0.48734377		
		0.31240946	0.410069446	0.4746190476	0.41617337	0.22684591	0.34361233	0.280196849	0.187994834	0.472067394	0.265501006	0.447902426	0.434060008	0.748461684	0.478461684	0.478461684		
Maria Consuelo Sánchez Diaz	Ronda 1	0.634904567	0	0	0	0	0	0.53244321	0.10896565	0.0512821	0.09910409	0.05991049	0.725	0.33333333	0.784611232	0.81481448		
		0.61844569	0	0	0	0	0	0.59090909	0.10909099	0	0.17909698	0.09386471	0.70526316	0.73800397	0.6721192	0.83229014		
		0.61095506	0	0	0	0	0	0.606969753	0.12328767	1	0	0	0.8652435	0.65829146	0.765167785	0.86486486		
Maria Consuelo Sánchez Diaz	Ronda 2	0.61438577	0	0	0	0	0	0.6134865	0.12857143	0	0.28481013	0.08158765	0.84418146	0.0869565	0.72	0.74193548		
		0.61610487	0	0	0	0	0	0.49315068	0.06779661	0.0578842	0	0	0.85434929	0.73737374	0.83116883	0.80740741		
		0.62219102	0	0	0	0	0	0.574904832	0.221157684	0.285543422	0.054912298	0.852615468	0.68924802	0.774997198	0.773625528	0.809625012		
José Luis Espinoza González	Ronda 1	0.58333333	0	0	0	0	0	0.65346553	0	0.0451613	0.28571429	0.052545444	0.80783636	0.82872928	0.82872928	0.82872928		
		0.58333333	0	0	0	0	0	0.65882233	0	0.05882233	0.28571429	0.052545444	0.80783636	0.82872928	0.82872928	0.82872928		
		0.57603093	0	0	0	0	0	0.62658967	0	0.05882233	0.28571429	0.052545444	0.80783636	0.82872928	0.82872928	0.82872928		
José Luis Espinoza González	Ronda 2	0.59514352	0	0	0	0	0	0.6791678	0.05882233	0.052545444	0.2	0.08652412	0.89410188	0.82341758	0.86712887	0.79754601		
		0.56072431	0	0	0	0	0	0.688	0	0.0212766	0.5	0.08688889	0.87919959	0.82608696	0.8	0.8555556		
		0.57662030	0	0	0	0	0	0.65544286	0.01	0.004034364	0.41469528	0.08001374	0.896760224	0.833679648	0.823658776	0.843551434		
Román Guadalupe de León Vázquez	Ronda 1	0.38738739	0	0.36818182	0.34871795	0.34361233	0.28706625	0	0.3255814	0.22098569	0.085542169	0.854615389	0.84615389	0.132948	0.9025641	0.93582888		
		0.385989212	0.39726027	0.348603922	0.35950413	0.26052632	0.5	0.2722727	0.2401071	0.2401071	0.2	0.8178539	0.90547264	0.91397849	0.81871783	0.81871783		
		0.41666667	0	0.39189189	0.38679245	0	0.21314952	0	0.18180188	0.24294186	0	0.25581395	0.82882883	0.81081081	0.84615388	0.88575706		
Hiram Efraim Orocio Garcia	Ronda 1	0.35691818	0.38926174	0.32888889	0.40609137	0.38974355	0.07476469	0	0.14861146	0.24294186	0.07476469	0.2877272	0.82310701	0.82309333	0.89050215	0.82852949		
		0.35691818	0.37062937	0.36322827	0.38541667	0	0.2302867	0	0.14285714	0.24294186	0.07476469	0.2877272	0.82310701	0.82309333	0.89050215	0.82852949		
		0.382194266	0.231430276	0.36004617	0.402610392	0.21857201	0.1	0.215902146	0.200156804	0.2857227	0.817324704	0.818943235	0.89227762	0.80994552				
Andrea Melissa Almeida Ortega	Ronda 1	0.81860792	0	0	0	0	0	0.84269663	0.06588682	0.08266667	0.0584992	0	0.89770911	0.53271028	0.568324532	0.33884298	0.28776712	
		0.80946801	0	0	0	0	0	0.82857143	0.06699635	0.09571788	0.061617361	0	0.90109755	0.49203649	0.61481481	0.4375	0.83896217	
		0.81626435	0	0	0	0	0	0.84061135	0.06744186	0.06727829	0.06064073	0	0.89358372	0.45669291	0.66666667	0.46153846	0.84459459	
Andrea Melissa Almeida Ortega	Ronda 2	0.80735688	0	0	0	0	0	0.83294664	0.05168269	0.08231707	0.06875934	0	0.88489583	0.480829126	0.56969552	0.57214285	0.84892086	
		0.81673307	0	0	0	0	0	0	0.83732922	0.05355064	0.01074419	0.06280742	0	0.88623075	0.48101266	0.54954955	0.4375	0.87692308
		0.81368643	0	0	0	0	0	0	0.8537054	0.05851054	0.05851054	0.068484208	0	0.89403337	0.48557426	0.59322208	0.49355064	0.847626364
Andrea Melissa Almeida Ortega	Ronda 3	0.40412776	0.37769597	0.45402299	0.52897471	0.49376441	0.29230809	0.38180089	0.38180089	0.38180089	0.38180089	0	0.89770911	0.53271028	0.568324532	0.33884298	0.28776712	
		0.4625	0.36649215	0.40740741	0.49432181	0.497476193	0.23140496	0.41285714	0.33233333	0.43181818	0.22580545	0.47561404	0.86597248	0.0947619	0.87845304	0.90196078		
		0.38129496	0.45333333	0	0	0	0	0.82050588	0.05755324	0.29354839	0.38461538	0.22720247	0.6875	0.20588235	0.88444444	0.89047319	0.96858639	0.97095579
Andrea Melissa Almeida Ortega	Ronda 4	0.40540541	0.36312849	0.421208905	0.504098361	0.53723404	0.22985075	0.66666667	0.32809524	0.3115942	0.21428571	0.89302326	0.02718447	0.9742268	0.92056075	0.96858639	0.97095579	
		0.3699187	0.38926174	0.32888889	0.40609137	0.38974355	0.19938335	0.2	0.5	0	0.20588235	0.82594118	0.9134529	0.923490727	0.943329897	0.98901099		
		0.404657148	0.389975056	0.231619451	0.483274846	0.483738078	0.240338488	0.337289376	0.3242242388	0.342297584	0.237093898	0.832581634	0.904950788	0.93723341	0.909955412	0.948874982		

Figura 10.12. Resultados de ejecución de los tres modelos sobre la primera evaluación de los primeros siete voluntarios.

11. Análisis e interpretación de resultados

El éxito es fácil de obtener. Lo difícil es merecerlo.
Albert Camus.

Tras la evaluación realizada en anteriores capítulos, los resultados se fueron registrando en una hoja de cálculo ordenando cuidadosamente los resultados de modo que sea posible comparar algunos de los resultados. Primeramente puede observarse en la figura 11.1 una comparación de eficiencia entre los tres algoritmos usando los conjuntos de datos en crudo de ambas evaluaciones, las de evocación de un concepto (rotuladas con el número 1) y las de intención de movimiento (etiquetadas con el número 2).

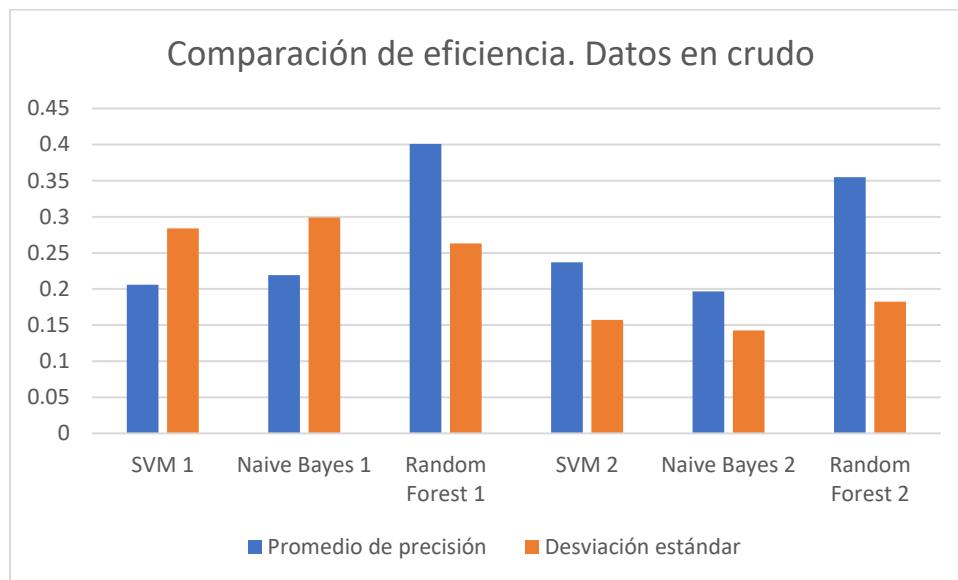


Figura 11.1. Comparación de eficiencia de los conjuntos de datos en crudo con ambas evaluaciones.

Puede observarse que en la evaluación de evocación de un concepto, SVM consigue una media de precisión de 0.20589154, muy similar a naive Bayes con una media de 0.21927877 de precisión, mientras que notablemente, random forest consigue una media de eficiencia de 0.40084800 siendo el mejor de los tres, aunque todos con valores de precisión muy bajos.

En la evaluación de intención del movimiento, la precisión media del SVM sube hasta un 0.23690072 mientras que naive Bayes baja a una media de 0.19646146 de precisión y random forest también decrece con respecto la evaluación anterior cayendo a 0.35477906 pero manteniéndose como el mejor de los tres en esta evaluación.

Es curioso observar que la dispersión de los datos es menor en la evaluación e intención de movimiento que en la de evocación de un concepto; esto observado a partir que las desviaciones estándar de los tres modelos son más bajas en la evaluación de intención del movimiento que en la de evocación de un concepto.

Dado que los valores de eficiencia son en sí muy bajos, la evaluación también se hizo con los conjuntos normalizados, obteniendo como resultado la gráfica que se visualiza en la figura 11.2 usando, en esta ocasión, los conjuntos de datos normalizados de ambas evaluaciones, nuevamente, las de evocación de un concepto rotuladas con el número 1 y las de intención de movimiento etiquetadas con el número 2.

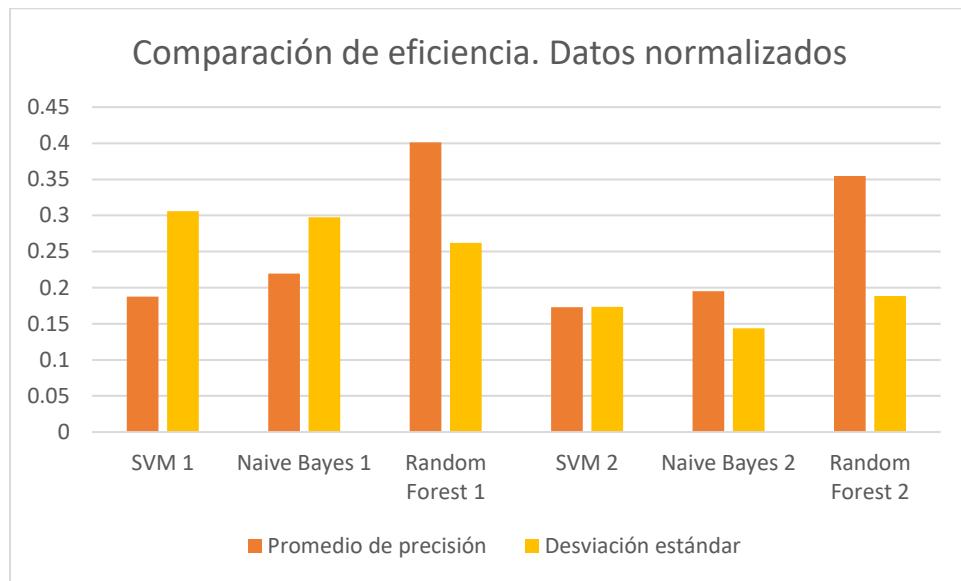


Figura 11.2. Comparación de eficiencia de los conjuntos de datos normalizados con ambas evaluaciones.

Nuevamente puede observarse que la SVM consigue una media de 0.18778087 de precisión, naive Bayes logra un 0.21953171 de efectividad media y random forest consigue 0.40139967 de precisión media en la evaluación de evocación de un concepto siendo este último nuevamente, el más alto de los tres.

Por otra parte, en intención del movimiento, SVM logra un puntaje de 0.17290407 medio de precisión, naive Bayes llega a 0.19509990 de precisión media y random forest alcanza un valor de 0.35450163 de precisión media que, al igual que con los datos en crudo, no mejora en comparación a la evaluación de evocación de un concepto pero sí se mantiene como el mejor de los tres algoritmos en intención del movimiento con datos normalizados.

La dispersión de los datos parece mantener un comportamiento similar a los datos en crudo con ciertas variaciones.

Sin embargo tenía que aplicársele el último paso del preprocesamiento de datos: la transformada de Fourier y con este se hizo el análisis definitivo evaluando los modelos en estas mismas condiciones. Los resultados se muestran en la figura 11.3 señalando nuevamente, las de evocación de un concepto rotuladas con el número 1 y las de intención de movimiento etiquetadas con el número 2.

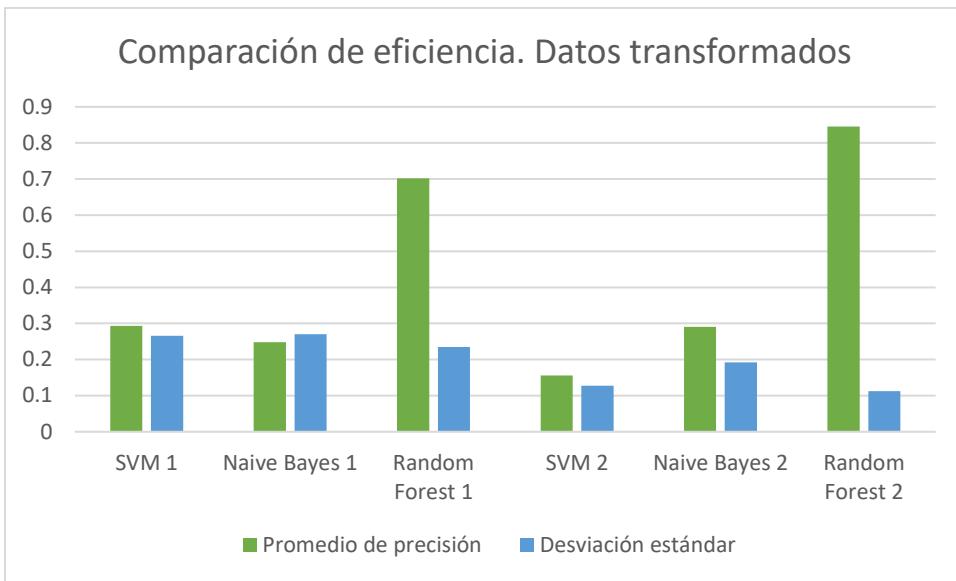


Figura 11.3. Comparación de eficiencia de los conjuntos de datos transformados con ambas evaluaciones.

He aquí la mayor sorpresa, pues nótese que SVM y naive Bayes logran un puntaje medio de precisión de 0.29375322 y 0.24820144 respectivamente, mientras que random forest, gracias a este paso del preprocesamiento de datos, sube su eficiencia media hasta un 0.70169729 en evocación de un concepto.

La tendencia es sorprendente en intención del movimiento, pues SVM baja su precisión 0.15635184 y naive Bayes consigue un valor de 0.29064879 de precisión media, pero nuevamente random forest obtiene el mejor puntaje medio de toda la experimentación con un récord de 0.84576270; incluso consigue la dispersión de datos más baja de todo el experimento también.

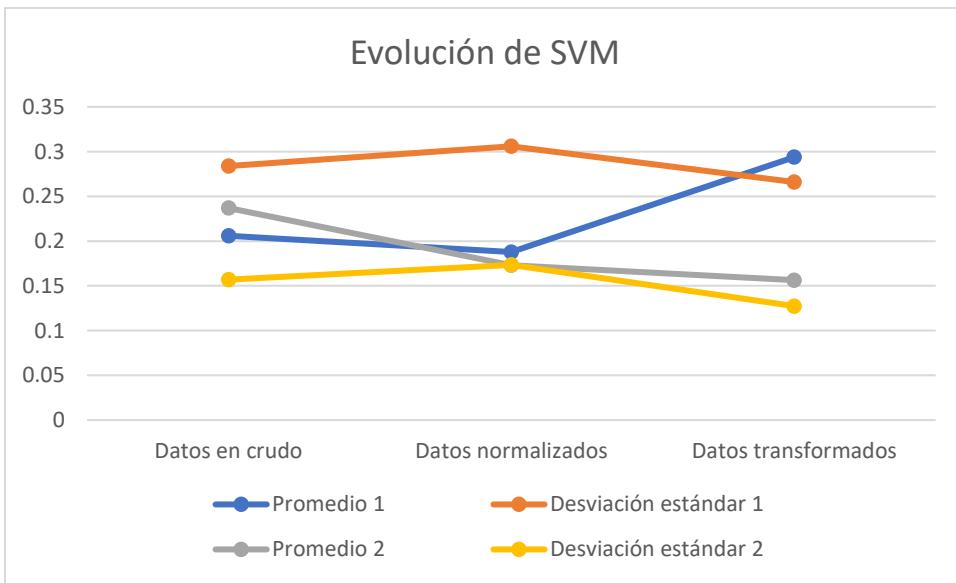


Figura 11.4. Evolución del SVM en ambas evaluaciones de procesos mentales.

Incluso es interesante pensar en cómo los modelos evolucionaron en estas tres etapas del preprocesamiento de datos y es lo que se analizará a continuación. Puede observarse en la figura 11.4 la evolución del modelo SVM en las tres etapas evaluadas, separando con el número 1 los valores de la evaluación de evocación de un concepto e intención de movimiento siendo etiquetados los valores con el número 2.

Puede observarse que la precisión media de evocación de un concepto mejora al final de las tres etapas, pero en intención de movimiento decae en cada uno de los pasos. La dispersión de datos parece mantener un comportamiento similar en las tres etapas.

Ahora, en la figura 11.5 se observa la evolución de naive Bayes señalando nuevamente, las gráficas de evocación de un concepto rotuladas con el número 1 y las de intención de movimiento etiquetadas con el número 2.

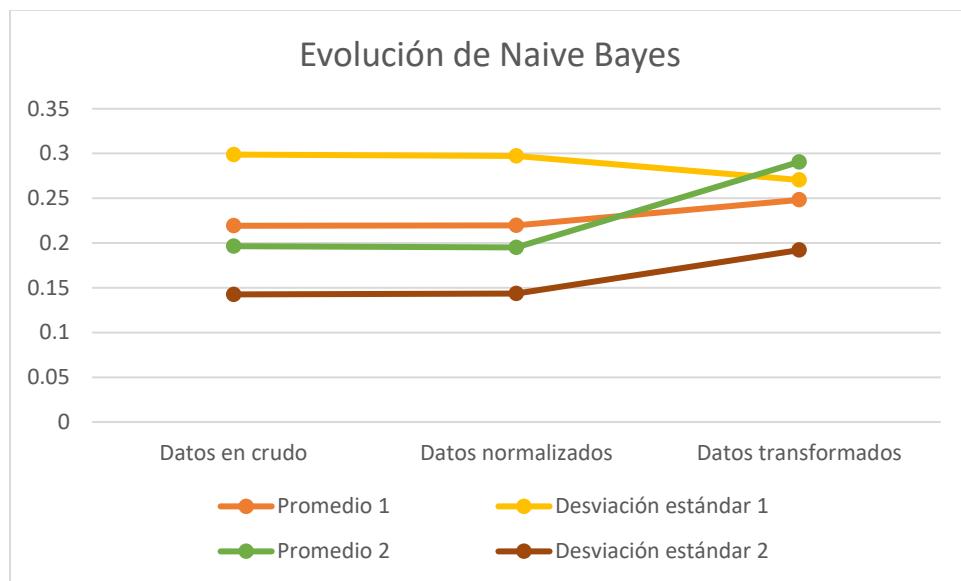


Figura 11.5. Evolución de naive Bayes en ambas evaluaciones de procesos mentales.

Aquí puede observarse que en ambos procesos mentales, al final de la evolución mejoran, siendo el de intención del movimiento el que consigue una mejora mayor que el de evocación de un concepto, aunque los valores de precisión no son tan buenos para ninguno de los dos procesos mentales en ninguna de las tres etapas.

La dispersión de los datos parece cerrarse de modo que en evocación de un concepto se presenta menor dispersión, mientras que en intención de movimiento la dispersión aumenta ligeramente a lo largo de la evolución del modelo.

Por último se observará la increíble evolución de random forest presentada en la figura 11.6 separando los datos de evocación de un concepto rotuladas con el número 1 y las de intención de movimiento etiquetadas con el número 2.

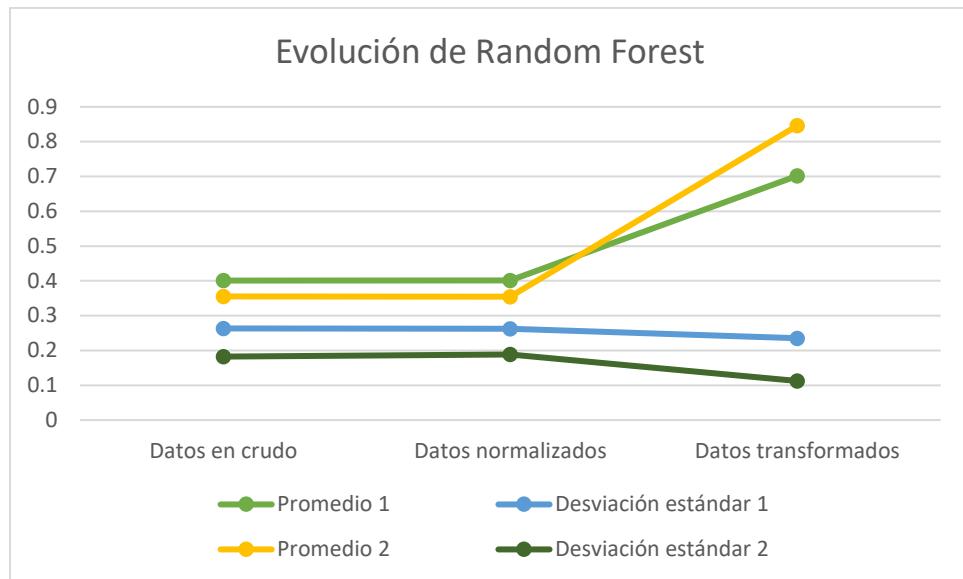


Figura 11.6. Evolución de random forest en ambas evaluaciones de procesos mentales.

Es sorprendente que ambos procesos maximicen la precisión y minimicen la dispersión. Claramente esto es más notorio en la evaluación de intención de movimiento pero consiguiendo en mayor medida esta optimización gracias a la última etapa, que es la de transformación de los datos.

Como gráficas adicionales, donde puede rescatarse la distribución de los datos – sobre todo para observar aquellos valores atípicos que no se pueden percibir en las gráficas previas – se exponen las gráficas siguientes, que son de caja y bigotes donde cada caja representa un concepto a predecir.

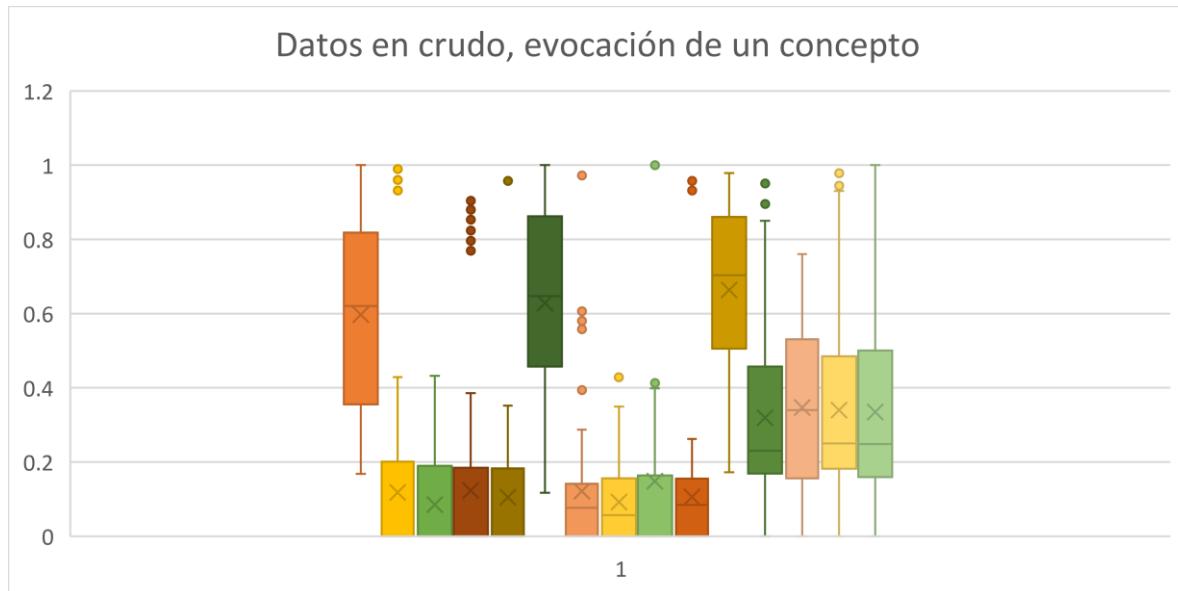


Figura 11.7. Gráfica de los resultados de ejecución de los tres modelos de machine learning usando los datos recopilados en la experimentación de campo usando la evaluación de evocación de un concepto de los voluntarios.

De izquierda a derecha están cinco cajas de los cinco conceptos a predecir por la SVM, después cinco cajas de los mismos cinco pero bajo naive Bayes y finalmente las cinco cajas correspondientes a los mismos conceptos bajo el modelo de random forest.

La evaluación de evocación de un concepto posee, de izquierda a derecha los conceptos árbol, computadora, cuaderno, perro y el control de no concentrarse en nada. La evaluación de intención de movimiento de izquierda a derecha abarca las intenciones de movimiento de mover el ratón o *mouse* de la computadora hacia arriba, abajo, izquierda, derecha y el control de no concentrarse en una intención específica.

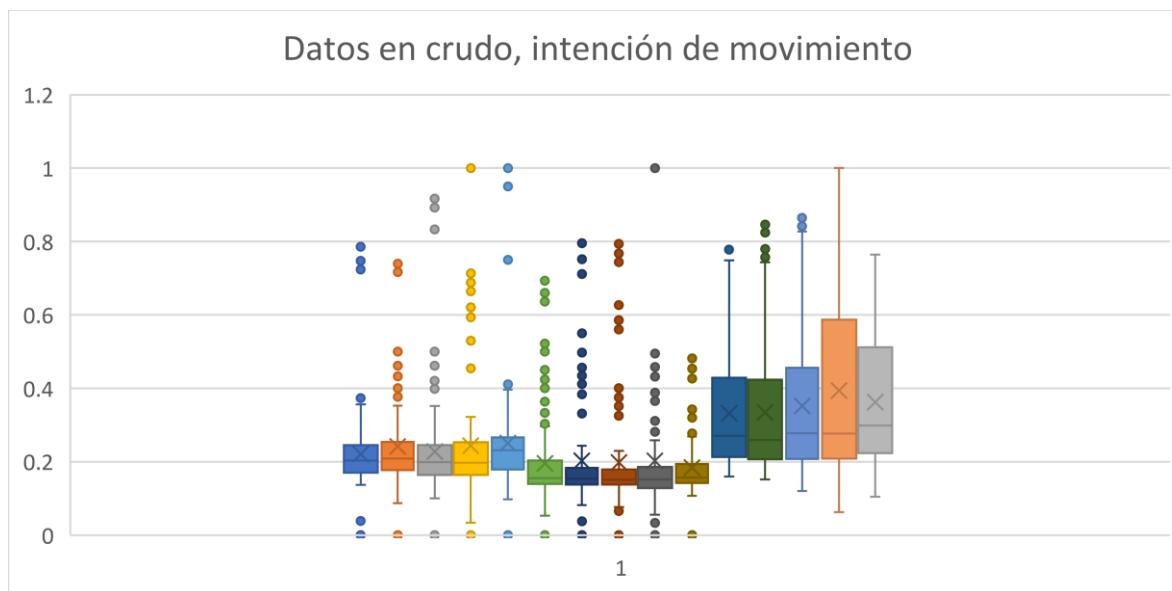


Figura 11.8. Gráfica de los resultados de ejecución de los tres modelos de machine learning usando los datos recopilados en la experimentación de campo usando la evaluación de intención de movimiento de los voluntarios.

Se pueden apreciar distintas interpretaciones de los resultados en crudo como los mencionados en gráficas previas, sin embargo, las figuras 11.7 y 11.8 refuerzan la conclusión que la precisión de los modelos no es fiable dado su bajo porcentaje de eficiencia alcanzado en general.

Ahora, tras el procedimiento de normalización nuevamente se pasaron los datos por los modelos para observar si existía algún cambio. Estos cambios se reflejan en las figuras 11.9 y 11.10.

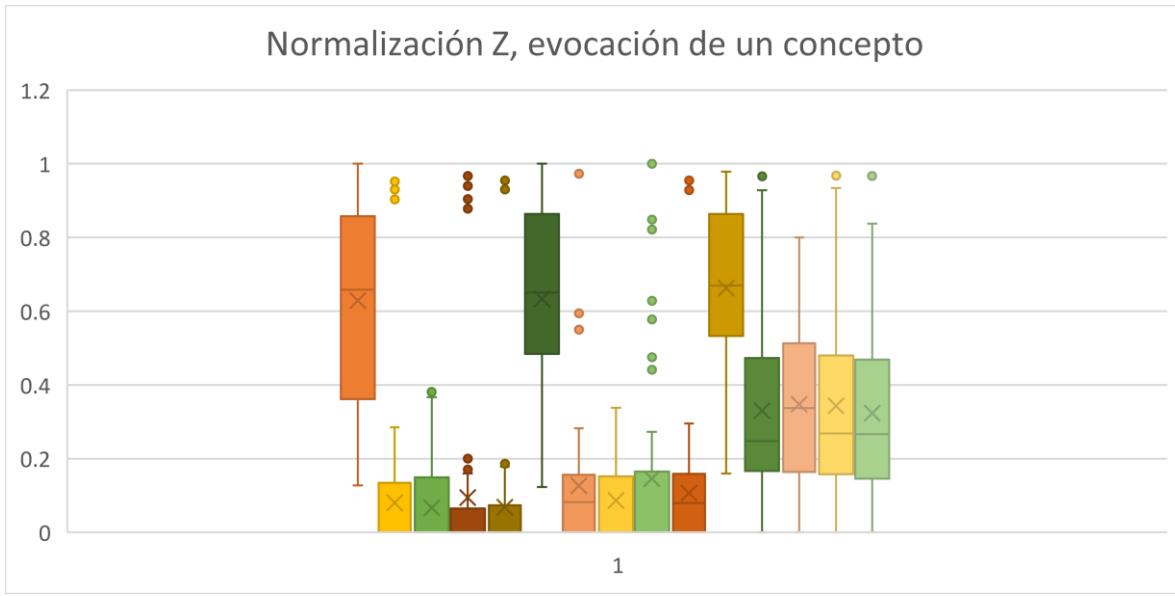


Figura 11.9. Gráfica de los resultados de ejecución de los tres modelos de machine learning usando los datos normalizados usando la evaluación de evocación de un concepto de los voluntarios.

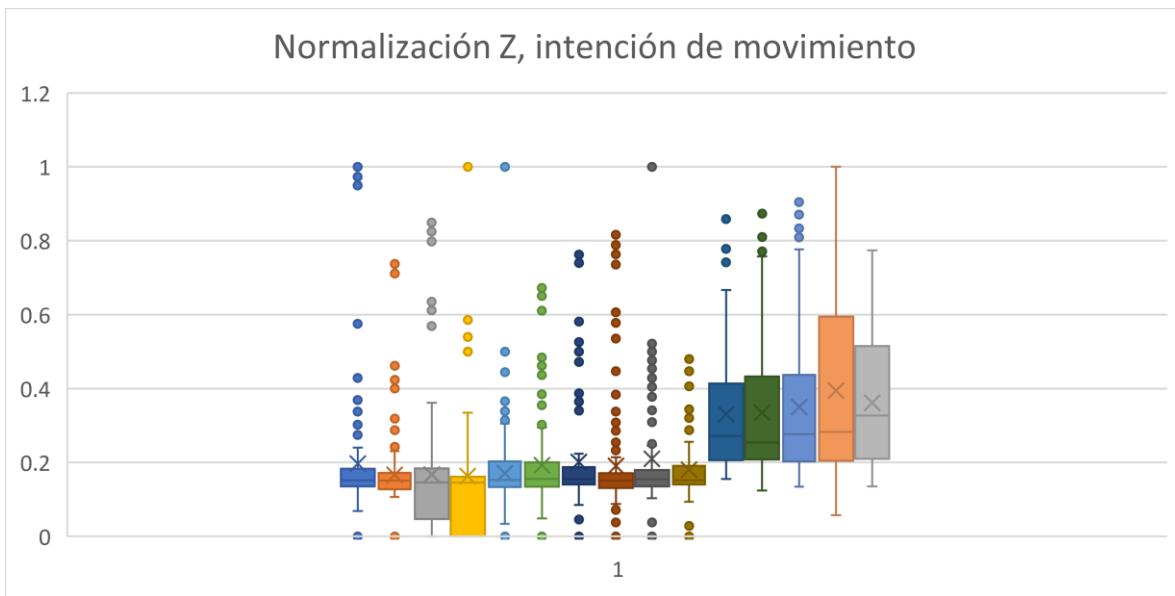


Figura 11.10. Gráfica de los resultados de ejecución de los tres modelos de machine learning usando los datos normalizados usando la evaluación e intención de movimiento de los voluntarios.

Pese a que los resultados no muestran en este caso una clara mejora o caída, la normalización brindaba una mejoría esperanzadora, pero no era suficiente. Es así que se avanza, como ya se describió previamente, a la transformación de Fourier. Los resultados se presentan en las figuras 11.11 y 11.12.

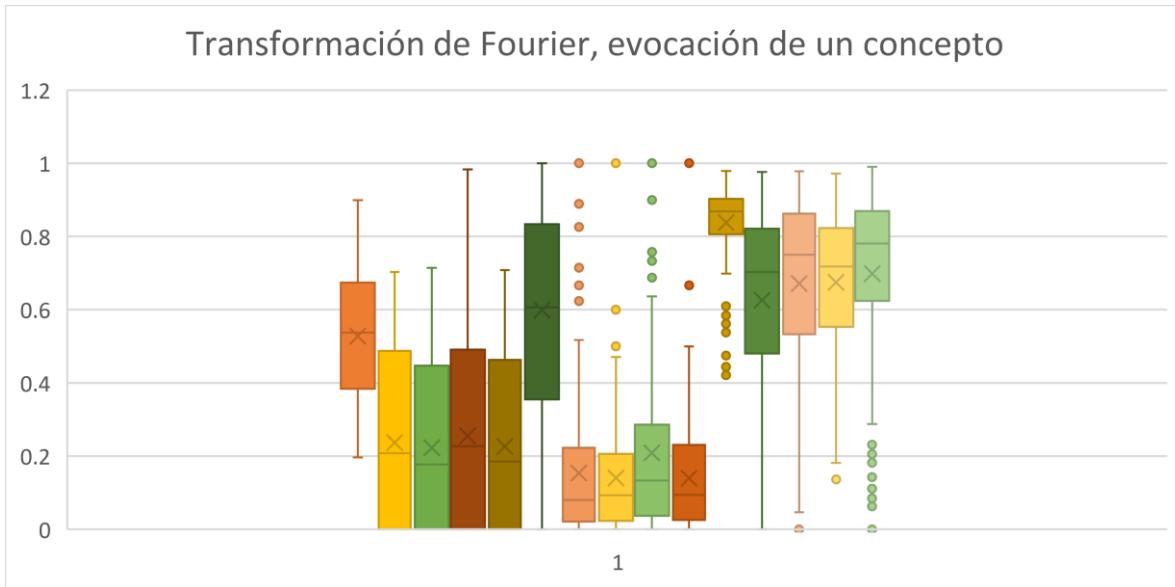


Figura 11.11. Gráfica de los resultados de ejecución de los tres modelos de machine learning usando los datos transformados mediante una transformada de Fourier usando la evaluación de evocación de un concepto de los voluntarios.

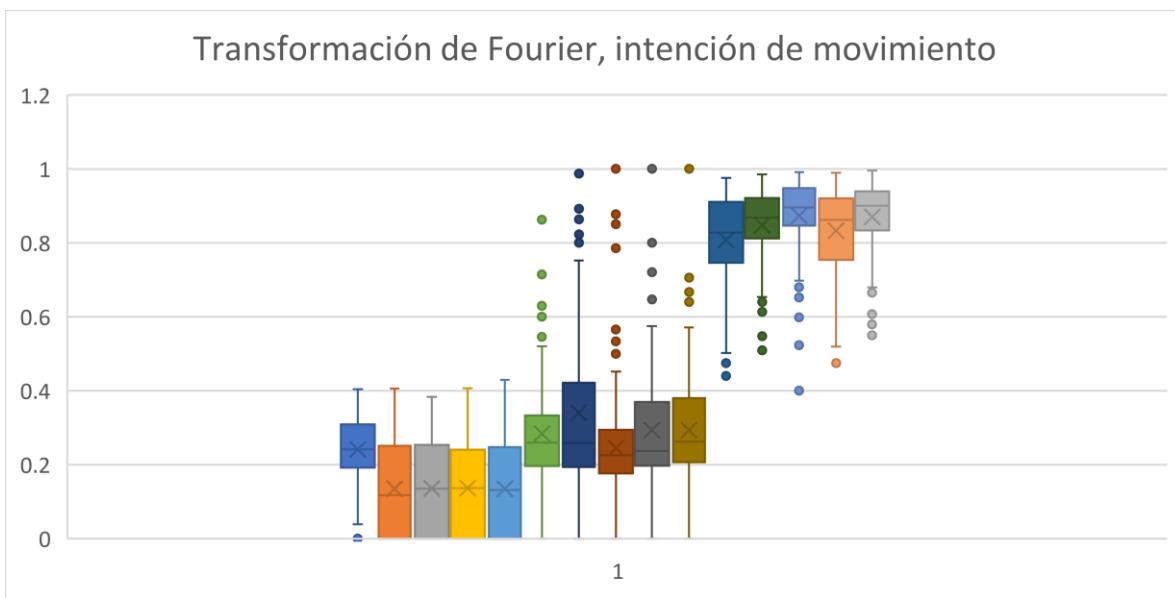


Figura 11.12. Gráfica de los resultados de ejecución de los tres modelos de machine learning usando los datos transformados mediante una transformada de Fourier usando la evaluación de intención de movimiento de los voluntarios.

Nuevamente y como en las gráficas previas, es aquí donde se percibe el mayor cambio, pues a partir de dicha transformación, los modelos escalan de peor a mejor siendo el que menor alcance tiene el SVM, siguiendo con naive Bayes que encuentra algunos valores atípicos en predicciones con alta fiabilidad pero estancándose aún en medias bajas y llegando al random forest que tiene una distribución muy aleatoria dentro de este procedimiento, con una distribución de predicciones cercana a valores que pueden considerarse mayormente fiables, aún con ciertas fallas, pero éstas

parecen ser más catalogadas como predicciones atípicas, de modo que con la transformación de Fourier puede observarse que de forma más eficiente el modelo de random forest obtiene resultados de predicción elevados posicionándose como el mejor modelo de entre los elegidos para esta investigación.

12. Conclusiones y discusión

Filosofar es esto: examinar y afinar los criterios.
Epicteto de Frigia.

La base de esta investigación yacía en la mejora de interfaces cerebro – computadora buscando con qué algoritmo de machine learning podría trabajarse mejor para realizar una interfaz de este tipo que fuera más efectiva. Con el uso de la diadema Emotiv Epoch+ junto con la paquetería de software mencionada además del código de Python desarrollado personalmente, se buscaba experimentar para conseguir datos de voluntarios quienes se concentrarían en pensamientos de evocación de un concepto y de principalmente intención de movimiento, así, usando los tres algoritmos de machine learning mencionados, entrenarlos con estos datos y observar la eficiencia que arrojan cada uno de estos.

A través del procedimiento se descubrieron muchas cosas como, primeramente que la hipótesis que señalaba personalmente donde el SVM sería el mejor algoritmo de entre los tres sería una hipótesis que no se cumplió al final del experimento, pues el algoritmo de random forest fue quien se llevó los mejores valores de precisión. Así también se descubrieron cuestiones como la importancia de pasos del preprocessamiento que en un principio no estaban contempladas, tales como la normalización y transformación de los datos.

Algunas líneas propuestas para continuar con esta investigación son, por ejemplo, arrancar con el modelo vencedor de la presente investigación buscando perfeccionar aspectos como la transmisión y clasificación en vivo, mientras los usuarios siguen usando la diadema o mejorar el porcentaje de efectividad y así realizar una implementación robusta de una interfaz cerebro – computadora tras el trabajo de esta y muchas investigaciones futuras. Otra línea de investigación es aumentar el conjunto de intenciones de movimiento para ampliar más acciones posibles que puedan realizarse como entrada a un equipo de cómputo.

También cabe recalcar que pueden existir puntos de mejora en la presente investigación que puede mejorarse en trabajo futuro, pues por ejemplo puede perfeccionarse el espacio y entorno de toma de datos con los voluntarios, perfeccionar las instrucciones de pensamientos deseados que se les dice a los voluntarios, buscar nuevos métodos de normalización o transformación y contrastarlos así como nuevos modelos de machine learning con los que explorar nuevas posibilidades o partir con lo presentado y avanzar en las mejoras recientemente propuestas pues el conocimiento presentado en esta investigación tiene distintas vertientes, todas igualmente aceptables de explorar e indagar sobre el conocimiento que en esos campos puedan encontrarse y aportar a esta área del conocimiento.

En relación a los objetivos de la investigación y la evaluación de éstos, comenzando por los objetivos particulares, se observa que la realización de una conexión eficiente entre el dispositivo Emotiv Epoch+ y un equipo computacional que permita extraer los datos de la actividad neuronal de un

usuario se concretó en su totalidad y de forma exitosa. Era más que necesario que así fuera, pues de lo contrario la investigación se estancaría y no podría avanzarse en lo presentado, de modo que su realización fue todo un éxito.

El objetivo que comprende la adaptación de las herramientas necesarias, tales como el programa desarrollado en Python por CymatiCorp: CyKit, el programa OpenViBE y el software de Emotiv, para que la recepción de datos del dispositivo que actúa como EEG se realice correctamente fue otro objetivo cumplido y satisfactoriamente, pues también el dominio y familiarización del software permitieron desarrollar cada vez de forma más cómoda la serie de pasos sucesiva que se desprendían al conseguir las conexiones entre todos estos programas nombrados.

La realización de preprocesamiento de los datos obtenidos en la experimentación de campo con los usuarios que colaboraron con registrar su actividad cerebral, pasando estos conjuntos de datos por técnicas de normalización y transformación que ayuden en una mejora del rendimiento de los modelos de machine learning fue otro objetivo cumplido satisfactoriamente. Hablando personalmente es el que me brindó mayor satisfacción de realizar y cumplir pues había olvidado conceptos de estadística, álgebra lineal, cálculo integral y variable compleja que son necesarios para entender e implementar los pasos de normalización y transformación. Volver a entender estos conceptos me llenó de satisfacción en la realización de esta exploración.

La implementación de los algoritmos support vector machine, random forest y naive Bayes para hacer las pruebas de aprendizaje sobre los datos de actividad neuronal también se concretó en su totalidad y de forma satisfactoria. En este apartado creo personalmente que pudo haberse explorado más, como las funciones kernel de las SVM pero por cuestión de tiempos se tuvo que realizar la investigación con una función kernel. Igualmente me hubiera gustado implementar uno o dos algoritmos de clasificación extra, sin embargo, el cumplimiento de este objetivo fue satisfactorio.

Como último objetivo específico, analizar cuál de los modelos previamente mencionados clasifica con mayor eficiencia la actividad neuronal del usuario en intención del movimiento que permitan operar una computadora se llevó a cabo en su totalidad completándose de forma grata ilustrando con distintas gráficas los puntos que se querían mencionar, así como visualizando mejor de esta forma los resultados obtenidos.

Ahora, hablando del objetivo que rige toda la investigación: comparar y encontrar el modelo de aprendizaje supervisado, entre el support vector machine, el modelo random forest y el modelo naive Bayes, cuál clasifica con mayor eficiencia datos obtenidos a partir de una diadema Emotiv Epoch orientado a la intención de movimiento usando un conjunto de acciones dados a los usuarios de prueba. El objetivo se da por terminado al concluir que el algoritmo random forest es el que mejor clasifica los datos obtenidos a partir de esta experimentación, viendo personalmente, una mejora sorprendente tras aplicarle una transformada de Fourier, creo yo, porque desprende una variable en

dos lo que la vuelve un análisis óptimo para el modelo de random forest y su naturaleza de operación basándose en árboles de decisión; lo cual también refuta la hipótesis realizada al inicio del capítulo, rechazando la conjectura de que SVM saldría victoriosa de este enfrentamiento de tres titanes, pues al final, el vencedor fue random forest.

Mencionando algunos aciertos y puntos de mejora, cabe destacar que como aciertos, considero ideal resaltar la exploración de OpenViBE pues, pese a que no encontré una forma de realizar el estudio con transmisión y clasificación de datos en vivo, esta indagación me permitió encontrar la forma de guardar los resultados en archivos separados por comas. Otro acierto fue gracias a los participantes de la investigación, ya que considero que se amasó una muestra muy fiable de datos con los que evaluar. Sin duda, el acierto clave de la investigación fue el sugerido por las directoras de esta investigación, quienes me comentaron la idea de realizar los procesos de normalización y transformación de datos que se realizaron en este trabajo.

Por otra parte, algunos puntos a mejorar de esta investigación pueden destacarse como uno muy clave, que fue una débil definición de instrucciones sobre el trabajo mental que debían realizar los voluntarios, pues esto debió definirse de mejor manera para evitar ambigüedades entre voluntarios. Otra cuestión que causaba esto que es motivo de mención es haber estandarizado un lugar con poco ruido, luz y personas alrededor de él, ya que los lugares variaban mucho, con personas que evaluaba a solas y otras que evaluaba con más compañeros y ellos mismos me comentaban que estaban nerviosos lo cual pudo haber afectado al conjunto de datos resultante.

13. Referencias

La definición de grandeza es inspirar a la gente cercana a ti. Así creas algo que nunca muere.
Kobe Bryant.

- Afifi, A., & Bergman, R. (1998). *Neuroanatomía funcional*. México, D.F.: McGraw Hill.
- Alpaydin, E. (2010). *Introduction to Machine Learning*. Cambridge: MA: MIT Press.
- Anderson, J. (1993). *Rules of the mind*. Londres: Psychology Press.
- APD, R. (2019, Abril 04). *¿Cuáles son los tipos de algoritmos del machine learning?* Récupéré sur APD: <https://www.apd.es/algoritmos-del-machine-learning/>
- Band, A. (2020, Mayo 9). *Multi-class Classification — One-vs-All & One-vs-One*. Récupéré sur Towards Data Science: <https://towardsdatascience.com/multi-class-classification-one-vs-all-one-vs-one-94daed32a87b>
- Barrett, K., Barman, S., Boitano, S., & Brooks, H. (2013). *Actividad eléctrica del cerebro, estados de sueño-vigilia y ritmos circadianos*. España: Access Medicine.
- Barsalou, L. (1999, Agosto 22). *Perceptual symbol systems*. Récupéré sur National Library of Medicine: <https://pubmed.ncbi.nlm.nih.gov/11301525/>
- Bayes, T. (1763). *An Essay towards solving a Problem in the Doctrine of Chances*. London: Philosophical Transactions of the Royal Society of London.
- Bear, M., Connors, B., & Paradiso, M. (2002). *Neurociencia: explorando el cerebro*. Barcelona: Masson.
- Betanzos Gómez, A. (2020, Marzo 13). *Principales algoritmos de Machine Learning*. Récupéré sur LinkedIn: <https://www.linkedin.com/pulse/principales-algoritmos-de-machine-learning-alejandro-betanzos-g%C3%B3mez/?originalSubdomain=es>
- Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. New York: Springer.
- Blog Unipython. (2018, Enero 15). *Support Vector Machines (SVM)*. Récupéré sur Unipython: <https://unipython.com/support-vector-machines-svm/>
- Brownlee, J. (27, Abril 2021). *One-vs-Rest and One-vs-One for Multi-Class Classification*. Récupéré sur Machine Learning Mastery: <https://machinelearningmastery.com/one-vs-rest-and-one-vs-one-for-multi-class-classification/>
- Caldaria. (2020, Julio 22). *Curiosidades sobre el cerebro*. Récupéré sur HDOSO Magazine: <https://www.caldaria.es/curiosidades-cerebro/>
- Cayre, M., Malaterre, J., Scotto-Lomassese, S., Strambi, C., & Strambi, A. (2010). The common properties of neurogenesis in the adult brain: from invertebrates to vertebrates. *Comparative Biochemistry and Physiology Part B: Biochemistry and Molecular Biology*, 1-15.
- Clinic, P. M. (2021, Junio 3). *Convulsiones del lóbulo frontal*. Récupéré sur Mayo Clinic: <https://www.mayoclinic.org/es-es/diseases-conditions/frontal-lobe-seizures/symptoms-causes/syc-20353958>
- Cromer, A. (1996). *Física para ciencias de la vida*. New York: Reverté.
- CymatiCorp. (2022). *CyKit*. Récupéré sur GitHub: <https://github.com/CymatiCorp/CyKit>

- Data Scientest. (2022, Enero 25). *Random Forest: Bosque aleatorio. Definición y funcionamiento*. Récupéré sur Data Scientest: <https://datascientest.com/es/random-forest-bosque-aleatorio-definicion-y-funcionamiento>
- Denby, B., Schultz, T., Honda, K., Hueber, T., Gilbert, J., & Brumberg, J. (2010). Silent Speech Interfaces. *Speech Communication*, 270-287.
- Ebbing, D., & Gammon, S. (2010). *Química General*. México, D.F.: Cengage Learning.
- El Estadístico Blogspot. (2021, Febrero 17). *Random Forest explicado de forma sencilla*. Récupéré sur Blog Estadístico: <https://elestatistico.blogspot.com/2021/02/random-forest-explicado-de-forma.html>
- Emotiv. (2022). *Emotiv*. Récupéré sur Emotiv: <https://www.emotiv.com/>
- Fadiga, L., Craighero, L., & Olivier, E. (2005, Abril 15). *Human motor cortex excitability during the perception of others' action*. Récupéré sur National Library of Medicine: <https://pubmed.ncbi.nlm.nih.gov/15831405/>
- Fisio, O. (2022, Septiembre 27). *¿Qué es soma o cuerpo neuronal?* Récupéré sur Fisioterapia Online: <https://www.fisioterapia-online.com/glosario/soma-o-cuerpo-neuronal>
- Flach, P. (2012). *Machine Learning: The Art and Science of Algorithms that Make Sense of Data*. Cambridge: Cambridge University Press.
- Flynn, K. (2013). The cytoskeleton and neurite initiation. *Bioarchitecture*, 86-109.
- González Barrio, H., Calleja Ochoa, A., Gómez-Escudero, G., Rodríguez Ezquerro, A., & López de Lacalle Marcaide, L. (2021, 04 06). *Los conceptos de Machine Learning y Deep Learning en la industria*. Récupéré sur Interempresas: <https://www.interempresas.net/MetalMecanica/Articulos/347471-Los-conceptos-de-Machine-Learning-y-Deep-Learning-en-la-industria.html>
- Gray, P. (2002). *Psychology*. New York: Worth Publishers.
- Green, P. (2008). *Iterative Design*. Michigan: Lecture presented in Industrial and Operations Engineering 436. University of Michigan.
- Herculano-Houzel, S. (2009). The human brain in numbers: a linearly scaled-up primate brain. *Hum Neurosci*.
- Hermann, N. (1997, Diciembre 22). *What is the function of the various brainwaves?* Récupéré sur Scientific American: <https://www.scientificamerican.com/article/what-is-the-function-of-t-1997-12-22/>
- Hill, R. (2006). *Fisiología Animal*. Bogotá: Médica Panamericana.
- Hodgkin, A., & Huxley, A. (1939). Action Potentials Recorded from Inside a Nerve Fibre. *Nature*, 710-711.
- Holmes, A., Illowsky, B., & Dean, S. (2022, Febrero 14). *La distribución normal estándar*. Récupéré sur OpenStax: <https://openstax.org/books/introducci%C3%B3n-estad%C3%ADstica-empresarial/pages/6-1-la-distribucion-normal-estandar>
- Huang, J. (2021, Diciembre). *Disfunción cerebral según su localización*. Récupéré sur Manual MSD: <https://www.msdmanuals.com/es-mx/hogar/enfermedades-cerebrales,-medulares-y-nerviosas/disfunci%C3%B3n-cerebral/disfunci%C3%B3n-cerebral-seg%C3%BAn-su-localizaci%C3%B3n>

- Huang, J. (2021, Octubre). *Generalidades sobre la función cerebral*. Récupéré sur Manual MSD: <https://www.msdmanuals.com/es-mx/professional/trastornos-neurol%C3%B3gicos/funciones-disfunciones/C3%20de-los-l%C3%ADbulos-cerebrales/generalidades-sobre-la-funci%C3%B3n-cerebral>
- Interpsiquis. (2022, Septiembre 25). *Lóbulo Límbico*. Récupéré sur Congreso Virtual de Psiquiatría: <https://psiquiatria.com/glosario/lobulo-limbico>
- Iranzo de Riquer, A. (2022, Abril 27). *¿Qué es un electroencefalograma?* Récupéré sur Clinc Barcelona: <https://www.clinicbarcelona.org/asistencia/pruebas-y-procedimientos/electroencefalograma>
- Joyanes Aguilar, L. (2008). *Fundamentos de Programación*. Madrid: McGraw Hill.
- Kandel, E., Schwartz, J., & Jessel, T. (2000). *Principles of Neural Science*. New York: McGraw Hill.
- Kaptelinin, V. (2012). Activity Theory. *Encyclopedia of Human-Computer Interaction*, Available online at http://www.interaction-design.org/encyclopedia/activity_theory.html.
- Kearney, D. (2019, Febrero 12). *HCI design for computer brain interfaces*. Récupéré sur Fluid Blog: <https://blog.fluidui.com/designing-brain-computer-interfaces/>
- Khan Academy Authors. (s.d.). *El plano complejo*. Récupéré sur Khan Academy: <https://es.khanacademy.org/math/algebra2/x2ec2f6f830c9fb89:complex/x2ec2f6f830c9fb89:complex-plane/a/the-complex-plane>
- Kolb, B. W. (2014). *Neuropsicología Humana*. Madrid: Médica Panamericana.
- Kole, M., & Stuart, G. (2012). Signal processing in the axon initial segment. *Neuron*, 235-247.
- Laguna, M. (2022, Marzo 17). *Lóbulo de la ínsula*. Récupéré sur Kenhub: <https://www.kenhub.com/es/library/anatomia-es/lobulo-de-la-insula>
- Latarjet, M., & Ruiz Liard, A. (2004). Encéfalo, Generalidades y Definición. *Anatomía Humana*, 168-169.
- Marín, A., Martínez, F., Ureña, L., & López, P. (2017, Septiembre 6). *El Lenguaje del Pensamiento*. Récupéré sur Ciencia Cognitiva: <https://www.cienciacognitiva.org/?p=1502>
- Marius, H. (2020, Junio 9). *Multiclass Classification with Support Vector Machines (SVM), Dual Problem and Kernel Functions*. Récupéré sur Towards Data Science: <https://towardsdatascience.com/multiclass-classification-with-support-vector-machines-svm-kernel-trick-kernel-functions-f9d5377d6f02>
- Marte, H. (2019). *Interfaces cerebro computador: Controlar cosas con la mente*. Récupéré sur Neuro Class: <https://neuro-class.com/cerebro-computador-controlar- cosas-con-la-mente/>
- Martínez Heras, J. (2020, Septiembre 29). *¿Clasificación o Regresión?* Récupéré sur IArtificial: <https://www.iartificial.net/clasificacion-o-regresion/>
- Martínez, E. (2022, Septiembre 28). *La ínsula qué es, dónde se encuentra y cuál es su función*. Récupéré sur PsicoActiva: <https://www.psicoactiva.com/blog/la-insula-que-es-donde-se-encuentra-y-cual-es-su-funcion/>
- MDurance. (2021, Septiembre 30). *Todo lo que debes saber sobre el potencial de acción*. Récupéré sur MDurance: <https://blog.mdurance.eu/academia/el-potencial-de-accion/>

- Merck & Co, I. (2022, Septiembre 27). *Estructura típica de una neurona*. Récupéré sur Manual MSD: <https://www.msdmanuals.com/es/hogar/multimedia/figure/estructura-t%C3%ADpica-de-una-neurona>
- Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. Cambridge: MA: MIT Press.
- Murre, J., & Sturdy, D. (1995). The connectivity of the brain: multi-level quantitative analysis. *Biological cybernetics*, 73.
- Nazareno, J. (2020, Junio 8). *Fascinación con el cerebro y lo neuro*. Récupéré sur Conexiones. Plataforma de Ciencias del Aprendizaje: https://thelearningsciences.com/fascinacion_cerebro_neuro/
- Oficina de comunicaciones NIH. (2019, Agosto 7). *¿Cómo beneficia la tecnología de rehabilitación a las personas con discapacidades?* Récupéré sur National Institute of Health: <https://espanol.nichd.nih.gov/salud/temas/rehabtech/informacion/ayuda>
- Ojeda Sahagún, J. L., & Icardo de la Escalera, J. M. (2004). *Neuroanatomía humana*. Barcelona: Masson.
- Olmo, M., Nave, A., & Nave, R. (2022, Octubre 5). *Action Potentials*. Récupéré sur Hyperphysics: <http://hyperphysics.phy-astr.gsu.edu/hbasees/Biology/actpot.html>
- OpenViBE Forum. (s.d.). *OpenViBE | Software for Brain Computer Interfaces and Real Time Neurosciences*. Récupéré sur OpenViBE Forum: <http://openvibe.inria.fr/>
- Orellana Alvear, J. (2018, Noviembre 16). *Árboles de decisión y Random Forest*. Récupéré sur Bookdown: <https://bookdown.org/content/2031/ensambladores-random-forest-parte-i.html>
- Palucci, P., Tabernig, C., Carrere, L., Tornero, A., Walter, J., Atum, Y., & Stahringer, G. (2023, Marzo 8). *Software de animación virtual comandado por una interfaz cerebro - computadora para rehabilitación cognitiva*. Récupéré sur Facultad de Ingeniería UNER: <http://biblioteca-fing.uner.edu.ar/cgi-bin/koha/opac-detail.pl?biblionumber=12957>
- Paniagua Soto, J. (2016). *Electroencefalograma (EEG)*. Récupéré sur Granada Neurofisiología: <https://www.granadanurofisiologia.com/neurofisiologia-clinica/electroencefalograma.html>
- Paniagua, R., Nistal, M., Sesma, P., Álvarez-Uría, M., Fraile, B., Anadón, R., & Sáez, F. (2002). *Citología e histología vegetal y animal*. España: McGraw Hill.
- Parrás, D., & Tedesco, A. (s.d.). *Probabilidad*. Buenos Aires: <https://estadisticadeluxemburgo.netlify.app/page4.html>.
- Parzen, E. (1987). *Teoría moderna de probabilidades y sus aplicaciones*. California: Limusa.
- Pelvig, D., Pakkenberg, H., Stark, A., & Pakkenberg, B. (2008). Neocortical glial cell numbers in human brains. *Neurobiology of aging*, 11.
- Personal Mayo Clinic. (2021, Junio 3). *Convulsiones del lóbulo frontal*. Récupéré sur Mayo Clinic: <https://www.mayoclinic.org/es-es/diseases-conditions/frontal-lobe-seizures/symptoms-causes/syc-20353958>
- Python Software Foundation. (2023). *Welcome to Python*. Récupéré sur Python: <https://www.python.org/>
- Quiroga Subirana, P. (2013, Julio 12). *¿Qué es el electroencefalograma?* Récupéré sur Top Doctors: <https://www.topdoctors.es/diccionario-medico/electroencefalograma>

- Ramos-Argüelles, F. M. (2009). *Técnicas básicas de electroencefalografía: principios y aplicaciones clínicas*. Pamplona: Servicio de Neurofisiología Clínica. Hospital Virgen del Camino.
- Randall, D., Burggren, W., & French, K. (1998). *Eckert Fisiología Animal*. Girona: McGraw Hill.
- Reyes Núñez, U., Soto Gómez, O., & Vicario Solórzano, C. (2022, Septiembre 1). *Interacción Humano - Computadora: Sus aplicaciones*. Récupéré sur IPN: Boletín UPIITA: <https://www.boletin.upiita.ipn.mx/index.php/ciencia/1013-cyt-numero-92/2085-interaccion-humano-computadora-sus-aplicaciones>
- Roche, H. (2003). *Lexikon Medizin*. Alemania: Urban & Schwarzenberg. Récupéré sur Urban & Schwarzenberg
- Roman, V. (2019, Abril 25). *Algoritmos Naive Bayes: Fundamentos e Implementación*. Récupéré sur Medium: <https://medium.com/datos-y-ciencia/algoritmos-naive-bayes-fundamentos-e-implementaci%C3%B3n-B3n-4bcb24b307f>
- Russell, S., & Norvig, P. (2004). *Inteligencia Artificial: Un Enfoque Moderno*. Madrid: Pearson.
- Sabater, V. (2020, Agosto 3). *Lóbulos cerebrales: características y funciones*. Récupéré sur La Mente Es Maravillosa: <https://lamenteesmaravillosa.com/lobulos-cerebrales-caracteristicas-yfunciones/>
- Sáenz, J. (2013). *Cálculo Vectorial*. Venezuela: Hipotenusa.
- Salvador, G., Ramírez-Gallego, S., Luengo, J., & Herrera, F. (2014). *Big Data: Preprocesamiento y calidad de datos*. Granada: Departamento de Ciencias de la Computación e Inteligencia Artificial, Universidad de Granada, España.
- Sanabria Castro, A. (2020, Mayo 19). *Una introducción a los Árboles de Decisión*. Récupéré sur Grupo Dabia: <https://www.grupodabia.com/post/2020-05-19-arbol-de-decision/>
- Scikit-learn's team development and maintenance. (2023). *Scikit-learn*. Récupéré sur Scikit-learn: <https://scikit-learn.org/stable/>
- Sheikh, A., & Korm, E. (1994). *Imagery in Sports and Physical Performance*. New York: Baywood Publishing Company.
- Singer, J. (2006). *Imagery in Psychotherapy*. Washington D.C.: American Psychological Association.
- Solé, R., & Manrubia, S. (1996). *Neurodinámica*. Barcelona: Edicions UPC.
- Sosa Romano, L. (2022, Septiembre 21). *Fisiología de la actividad eléctrica del cerebro: electroencefalografía*. Récupéré sur Departamento de Fisiología. UNAM.: <https://fisiologia.facmed.unam.mx/index.php/category/unidad-tematica-i/>
- Sruthi, E. R. (2022, Noviembre 30). *Understanding Random Forest*. Récupéré sur Analytics Vidhya: <https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/>
- Sweigart, A. (2022). *Welcome to PyAutoGUI's documentation*. Récupéré sur GitHub Documentation: <https://pyautogui.readthedocs.io/en/latest/>
- The MathWorks Inc. (2022). *Diagrama de Bode de respuesta en frecuencia o datos de magnitud y fase*. Récupéré sur MathWorks: https://www.mathworks.com/help/control/ref/lti.bode_es.html

- The MathWorks Inc. (s.d.). *Support Vector Machine (SVM)*. Récupéré sur MathWorks: <https://la.mathworks.com/discovery/support-vector-machine.html>
- TIBCO Data Science. (2022, Septiembre 12). *¿Qué es el aprendizaje supervisado?* Récupéré sur TIBCO: <https://www.tibco.com/es/reference-center/what-is-supervised-learning>
- Tidwell, J. (1999, Mayo 17). *A Pattern Language for Human-Computer Interface Design*. Récupéré sur MIT Education: http://www.mit.edu/~jtidwell/common_ground_onefile.html
- Torres-García, A. A., Reyes-García, C. A., Villaseñor-Pineda, L., & Ramírez-Cortés, J. (2013). Análisis de señales electroencefalográficas para la clasificación de habla imaginada. *Revista mexicana de ingeniería biomédica*, 23-39.
- Triglia, A. (2015, Agosto 25). *Los 5 lóbulos del cerebro y sus distintas funciones*. Récupéré sur Psicología y Mente: <https://psicologiyamente.com/neurociencias/lobulos-del-cerebro-funciones>
- Triglia, A. (2016, Agosto 23). *Sistema límbico: la parte emocional del cerebro*. Récupéré sur Psicología y Mente: <https://psicologiyamente.com/neurociencias/sistema-limbico-cerebro>
- Urrestarazu, E. (2022). *Electroencefalograma*. Récupéré sur Clínica Universidad de Navarra: <https://www.cun.es/enfermedades-tratamientos/pruebas-diagnosticas/electroencefalograma>
- von Bartheld, C., Bahney, J., & Herculano-Houzel, S. (2016). The search for true numbers of neurons and glial cells in the human brain: A review of 150 years of cell counting. *The Journal of Comparative Neurology*, 18.
- Walpole, R., Myers, R., Myers, S., & Ye, K. (2012). *Probabilidad y Estadística para Ingeniería y Ciencias*. San Antonio: Pearson.
- Wikipedia. (2022, Octubre 1). *Neurona*. Récupéré sur Wikipedia: <https://es.wikipedia.org/wiki/Neurona>
- Wikipedia. (2022, Noviembre 8). *Ondas cerebrales*. Récupéré sur Wikipedia: https://es.wikipedia.org/wiki/Ondas_cerebrales
- Wikipedia. (s.d.). *Arcotangente*. Récupéré sur Wikipedia: <https://es.wikipedia.org/wiki/Arcotangente>
- Wikipedia. (s.d.). *atan2*. Récupéré sur Wikipedia: <https://en.wikipedia.org/wiki/Atan2>
- Wikipedia. (s.d.). *Electroencefalografía*. Récupéré sur Wikipedia: <https://es.wikipedia.org/wiki/Electroencefalograf%C3%ADA>
- Wikipedia. (s.d.). *Interacción persona-computadora*. Récupéré sur Wikipedia: https://es.wikipedia.org/wiki/Interacci%C3%B3n_persona-computadora
- Wikipedia. (s.d.). *Interfaz cerebro - computadora*. Récupéré sur Wikipedia: https://es.wikipedia.org/wiki/Interfaz_cerebro-computadora
- Wikipedia. (s.d.). *Transformada de Fourier*. Récupéré sur Wikipedia: https://es.wikipedia.org/wiki/Transformada_de_Fourier
- World Health Organization. (2023). *Disability*. Récupéré sur World Health Organization: https://www.who.int/health-topics/disability#tab=tab_1

14. Anexos

14.1. Anexo 1: Configuración para recibir datos desde la diadema hacia el equipo de cómputo usando la paquetería de software de Emotiv, OpenViBE, y la librería de Python: Cykit.

Primeramente, dado que este proyecto está usando la diadema de la marca Emotiv, será necesario instalar el software de esta marca. La paquetería de programas puede encontrarse – a la fecha de redacción de esta investigación – en la siguiente liga la cual es la página de descargas del software. De la misma forma, en la figura 14.1.1.

<https://www.emotiv.com/emotiv-launcher/>

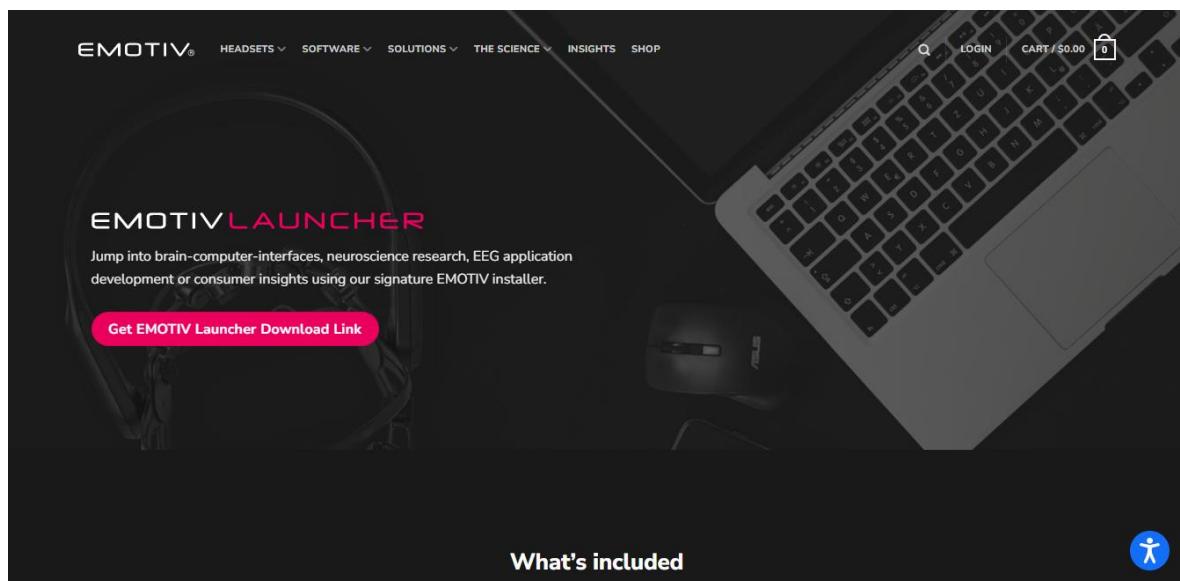


Figura 14.1.1. Sitio web de Emotiv.

En el sitio web anterior se descargará e instalará el paquete cuyo nombre es “Emotiv Launcher”. Durante el proceso de instalación se preguntará por el conjunto de programas a instalar, entre los cuales están “Emotiv Launcher”, “Emotiv BrainViz”, “EmotivPRO” y “EmotivBCI”. Como elección personal, se decidió instalar todo el conjunto de programas ofrecidos durante la instalación. Para poder usar los programas es necesario registrarse en la página de Emotiv y crear lo que la compañía llama un “Emotiv ID” que es una cuenta propia dentro de su sitio web. Esto puede hacerse desde la siguiente liga.

<https://www.emotiv.com/my-account/>

Asimismo, se muestra en la figura 14.1.2 el ejemplo visual del sitio web en el que hay que realizar el registro de la cuenta Emotiv para poder hacer uso del software de esta empresa.

Create Your EmotivID

Fields with (*) are required.

EmotivID *

Password *

Password must:

- be at least 8 characters long.
- contain at least one upper and one lowercase.
- not equal to username or email.

Confirm Your Password *

Email *

Activation email will be sent to this address.

We will also grab your [Gravatar](#) from this address.

Figura 14.1.2. Registro para la cuenta Emotiv ID.

Una vez completada la instalación, el programa Emotiv Launcher será el encargado de detectar si una diadema Emotiv está conectada al equipo. Puede observarse en la figura 14.1.3 este programa antes de conectar una diadema. Pero para conectar la diadema, es necesario – al menos en el caso del modelo EPOC+ – encender la diadema y conectar el dispositivo USB Bluetooth que viene con el paquete de la diadema tal como se aprecia en la figura 14.1.4.

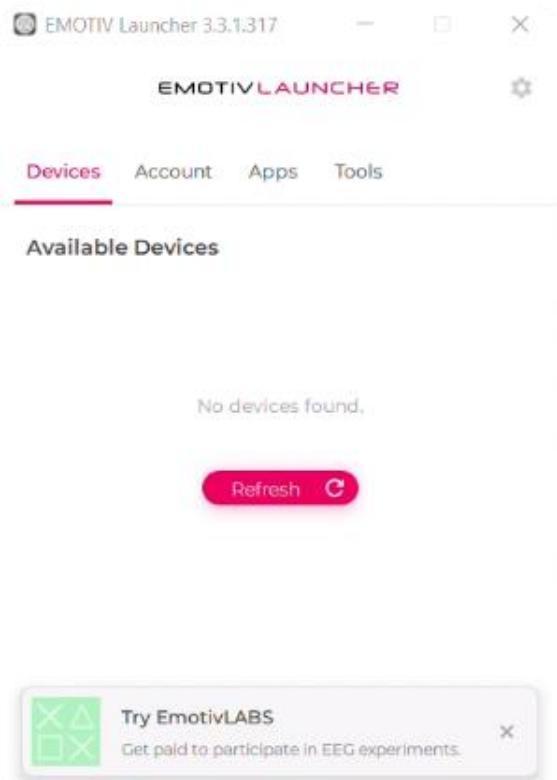


Figura 14.1.3. Emotiv Launcher sin la detección de una diadema conectada al equipo.



Figura 14.1.4. Diadema Emotiv encendida y dispositivo USB conectado al equipo.

Una vez que la conexión se haya establecido correctamente, Emotiv Launcher mostrará la diadema que detecta y permitiéndole al usuario conectarse a ella desde la interfaz de su programa, tal y como se muestra en la figura 14.1.5.

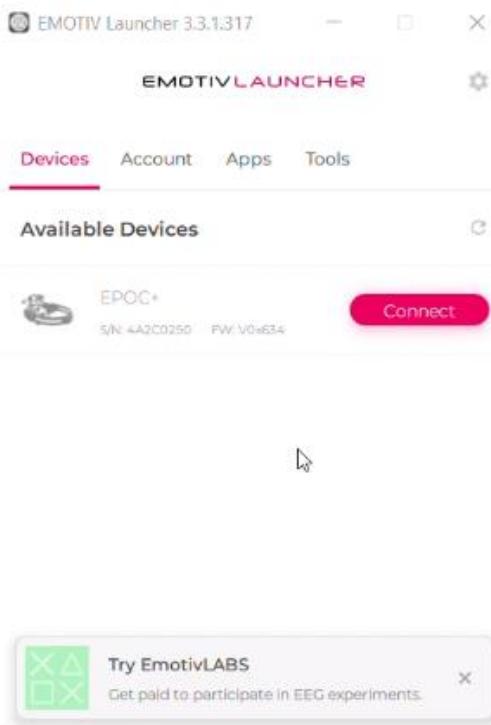


Figura 14.1.5. Emotiv Launcher detectando una diadema conectada al equipo.

A continuación, es necesario que el usuario en cuestión acomode la diadema sobre su cabeza de la manera correcta, realizando un acomodo que permita tanto una gran calidad de contacto y de recepción de las señales EEG. La figura 14.1.6 muestra este procedimiento guiado por la interfaz de Emotiv Launcher.

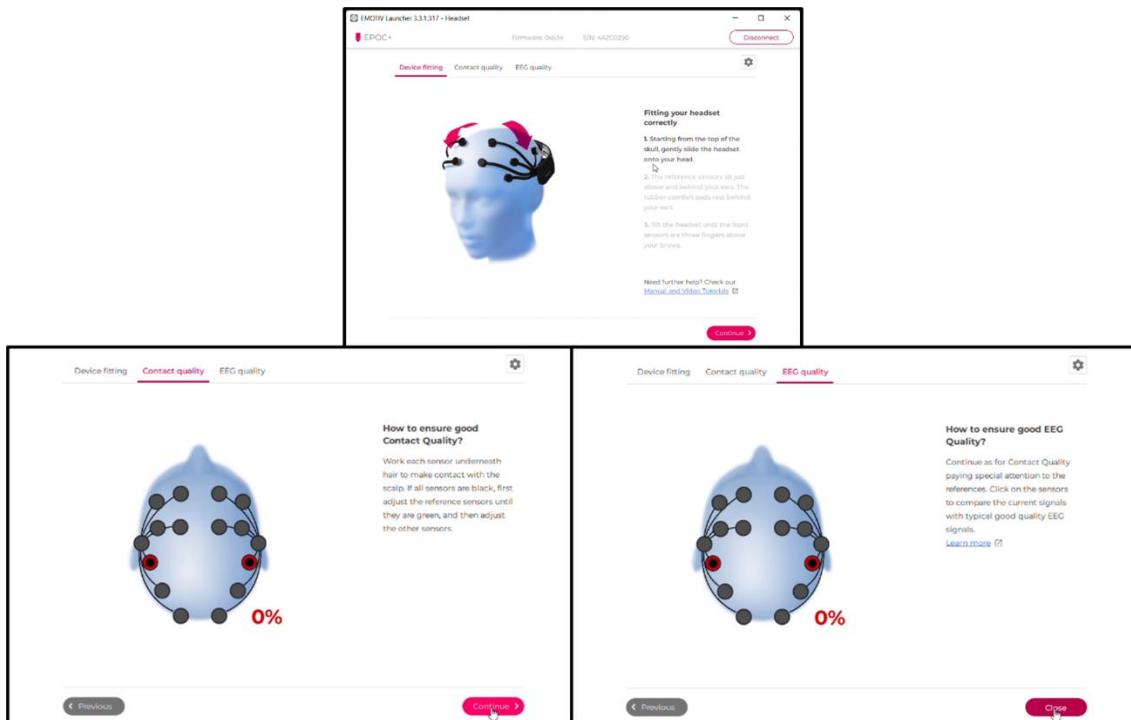


Figura 14.1.6. Emotiv Launcher guiando al usuario a acomodar la diadema.

Una vez que esté correctamente configurada la diadema con el equipo puede apagarse para un posterior uso. Es necesario también instalar el repositorio de GitHub creado por CymatiCorp: CyKit el cual puede encontrarse en la siguiente liga:

<https://github.com/CymatiCorp/CyKit>

The GitHub repository page for CyKit. At the top, there's a search bar, navigation links for Pull requests, Issues, Marketplace, and Explore, and user stats for Watch (12), Fork (37), and Star (114). Below that, there are tabs for Code, Issues (16), Pull requests (1), Actions, Projects, Wiki, Security, and Insights. The Code tab is selected, showing the master branch with 2 branches and 0 tags. A list of recent commits is displayed, all made by warrenarea. The commits are: 'Fixed erroneous encapsulation of bluetooth functions.' (4ee1762, 17 days ago, 369 commits), 'Create example_insight.py' (2 years ago), 'Fixed erroneous encapsulation of bluetooth functions.' (17 days ago), 'Added model 7 (EPOC+ in 14-bit mode) for bluetooth.' (28 days ago), 'LICENSE' (5 years ago), and 'Update README.md' (4 months ago). Below the code area, there's a 'CyKit' logo. To the right, there's an 'About' section with a brief description of the Python 3x server, links to the GitHub wiki, and social sharing options. The 'Releases' section indicates 'No releases published'.

Figura 14.1.7. Sitio web de GitHub donde se encuentra el repositorio del proyecto de CyKit por CymatiCorp.

El sitio web puede observarse en la figura 14.1.7. Desde este sitio web puede accederse a la documentación para instalar este paquete y usarlo junto a una herramienta denominada OpenViBE para producir información a partir de la diadema. Las dos siguientes ligas corresponden a cada apartado de la documentación.

<https://github.com/CymatiCorp/CyKit/wiki/How-to-Install-CyKIT>

<https://github.com/CymatiCorp/CyKit/wiki/How-to-Stream-Data-to-OpenViBE>

Se tomará estrictamente lo que dicen ambas guías a continuación para poder avanzar con las instalaciones pertinentes.

Como, inicialmente se debe hacer, habrá que instalar primeramente la librería CyKit; para ello será necesario contar con una versión de Python, ya sea la 3.7.2 o superior. En el caso propio se trabajó con la versión 3.10.

Posteriormente hay que descargar el repositorio. Esto puede hacerse de varias formas, ya sea descargando el archivo ZIP con todo el contenido del repositorio o usando la tecnología Git – que será el caso propio – clonando el repositorio en el directorio del gusto personal usando el comando de la figura 14.1.8.

```
1 git clone https://github.com/CymatiCorp/CyKit.git
```

Figura 14.1.8. Comando de clonación del repositorio de CyKit.

A continuación, puede probarse si la instalación se realizó correctamente probando el programa. Para ello, en una consola que esté ubicada en la raíz del directorio en donde se descargó el repositorio se ejecutarían los comandos de la figura 14.1.9.

```
1 cd CyKit-master  
2 cd Py3  
3 python CyKIT.py
```

Figura 14.1.9. Comandos de ejecución de CyKit.

```

The pairing name can easily be found in Windows Bluetooth settings.

Join these options (in any order), using a + separator.

(e.g info+confirm)

Example Usage:
Python.exe .\CyKIT.py 127.0.0.1 54123 1 info+confirm

Example Usage:
Python.exe .\CyKIT.py 127.0.0.1 5555 6 openvibe+generic+nocounter+noheader+nobattery+ovdelay:100+integer+ovsamples:004

> USB Device (No Additional Information)
> No USB Device Available. Exiting . . .

C:\Users\Propietario\Desktop\Cykit\CyKit\Py3>

```

Figura 14.1.10. Ejecución correcta de CyKIT.py usando Python.

En la figura 14.1.10 puede observarse la ejecución correcta del archivo CyKIT.py usando Python y por consecuencia, una correcta instalación de la paquetería. Ahora que ya está instalado el paquete de Python, para transmitir datos será necesario ejecutar el archivo similar a como anteriormente se realizó, pero deben de tomarse algunas consideraciones dependiendo de la diadema, frecuencia de transmisión, entre otras cosas, las cuales menciona CymatiCorp en la explicación de instalación y que puede apreciarse en la figura 14.1.11 y en la figura 14.1.12.

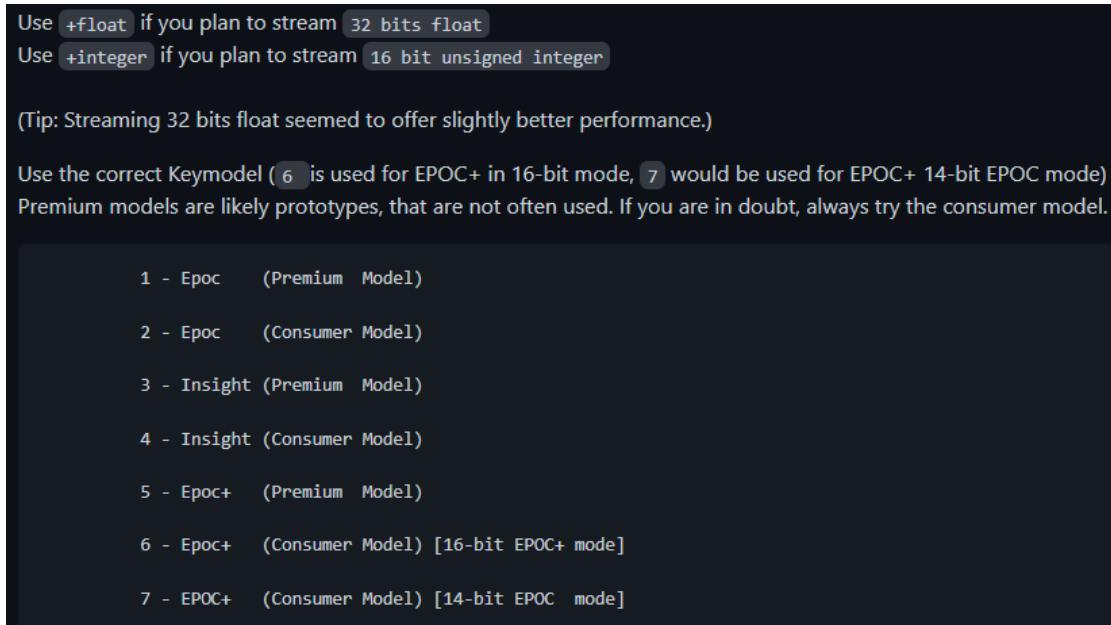


Figura 14.1.11. Explicación de algunas variables de configuración para ejecutar CyKit.

`+openvibe` This option indicates an OpenViBE data stream will be sent in a format readable by the `Generic Raw Telnet Reader` format. (More information in Step 5.)

`+generic` This option indicates a generic TCP stream has been created.

Mainly used to differentiate between a generic TCP stream, that anything can connect to and between a TCP stream that may be used by an HTTP web socket. (Requiring a handshake.)

`+nocounter` This option is used to remove the first two bytes of data that is typically sent in the data stream. Where the two bytes are `[COUNTER] [INTERPOLATE]`

`[COUNTER]` being the EEG packet number 0 - 127 (or 0 - 255 for 256hz mode)

`[INTERPOLATE]` being the number `16` for EEG data, and `32` to indicate data is gyro data.

By default, this option enables `+eegmode` which will send only EEG data, (ignoring Gyro data which could potentially corrupt the stream of data.) It sets this to EEG data by default, because otherwise there would be no way to indicate if the data stream is for EEG data, or Gyro data.

`+noheader` by default, CyKIT will send header information to both "Generic" TCP servers, and a websocket server. This header information includes various useful information about what config settings were set to run the CyKIT server. This header information is passed along, so that other programs can make use of the various settings available to CyKIT.

The header information is not however useful to programs like OpenViBE, so we include this config flag to disable sending this extra info.

`+nobattery` by default, the battery and quality settings are mixed into the data.

CyKIT sorts this data and sends it via two bytes, which is appended to the EEG data packet.

The battery and quality information are not in any format recognizable by OpenViBE, so this data should be removed to satisfy the OpenViBE format.

`+ovdelay:NNN` is a custom delay counter, designed to work in milliseconds, but results may vary by CPU speed. Where NNN is a number between 001 and 999. This delay will occur before each packet is sent to OpenViBE.

`+float` determines the format the microvoltage data will be sent in. (32-bit big-endian float) "Big Endian"

`+integer` determines the format the microvoltage data will be sent in. (16-bit unsigned integer) "Big Endian"

`+ovsamples:NNN` defines how many samples will be collected in a block of data, before sending to the openViBE server. (This will relate to Step 5. determining `Sample counter per sent block`)

Default for ovsamples is 004.

CyKIT can set the sample block to any number between 001 and 999. However OpenViBE only allows specific block sizes.

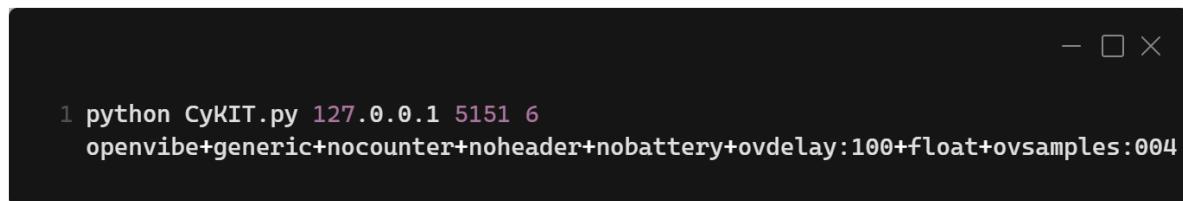
OpenViBE allows for the following sample block sizes: `004, 016, 032, 064, 128, 256, 512`

Where the larger the packet block is, the longer it will take to be processed by OpenViBE functions.

(See below, for altering OpenViBE to accept 001 packet samples, for faster stream processing.)

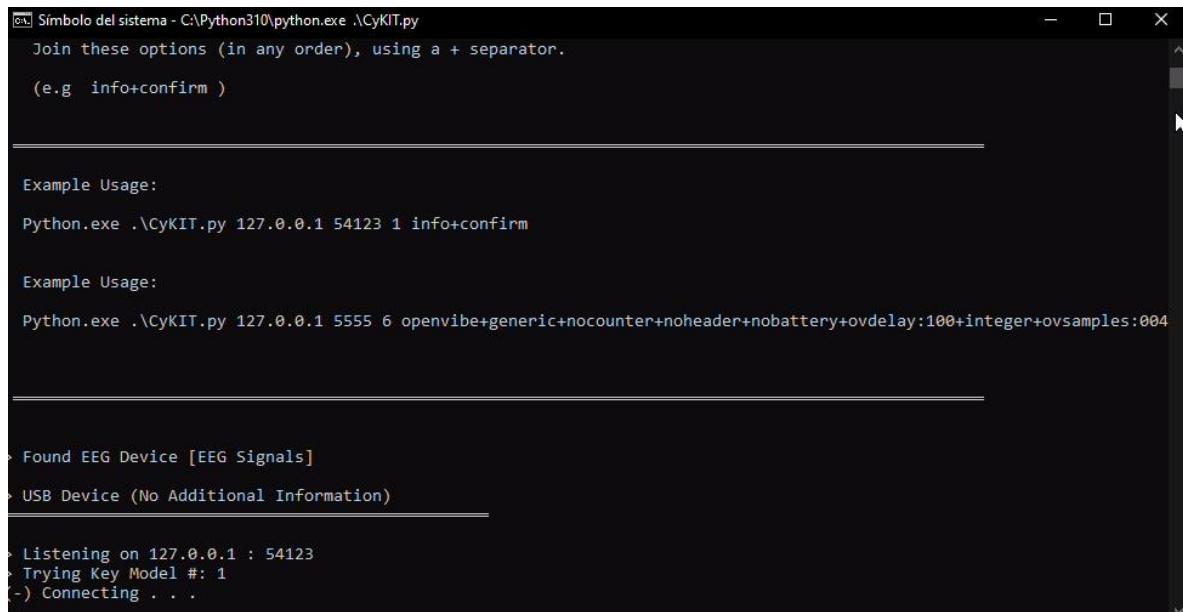
Figura 14.1.12. Explicación de algunas variables de configuración para ejecutar CyKit.

En el caso propio, tomando en cuenta las configuraciones recomendadas por CymatiCorp, se eligió por ejecutar el comando de la figura 14.1.13.



```
1 python CyKIT.py 127.0.0.1 5151 6
openvibe+generic+nocounter+noheader+nobattery+ovdelay:100+float+ovsamples:004
```

Figura 14.1.13. Comando de ejecución personalizada de CyKit.



```
Símbolo del sistema - C:\Python310\python.exe .\CyKIT.py
Join these options (in any order), using a + separator.
(e.g. info+confirm)

Example Usage:
Python.exe .\CyKIT.py 127.0.0.1 54123 1 info+confirm

Example Usage:
Python.exe .\CyKIT.py 127.0.0.1 5555 6 openvibe+generic+nocounter+noheader+nobattery+ovdelay:100+integer+ovsamples:004

> Found EEG Device [EEG Signals]
> USB Device (No Additional Information)
> Listening on 127.0.0.1 : 54123
> Trying Key Model #: 1
(-) Connecting . . .
```

Figura 14.1.14. CyKit ejecutándose bajo las configuraciones indicadas.

En la figura 14.1.14 puede observarse el correcto funcionamiento de CyKit al calibrarse con las opciones especificadas. Ahora, el siguiente paso será instalar OpenViBE, el cual se descarga desde la siguiente liga:

<http://openvibe.inria.fr/downloads/>

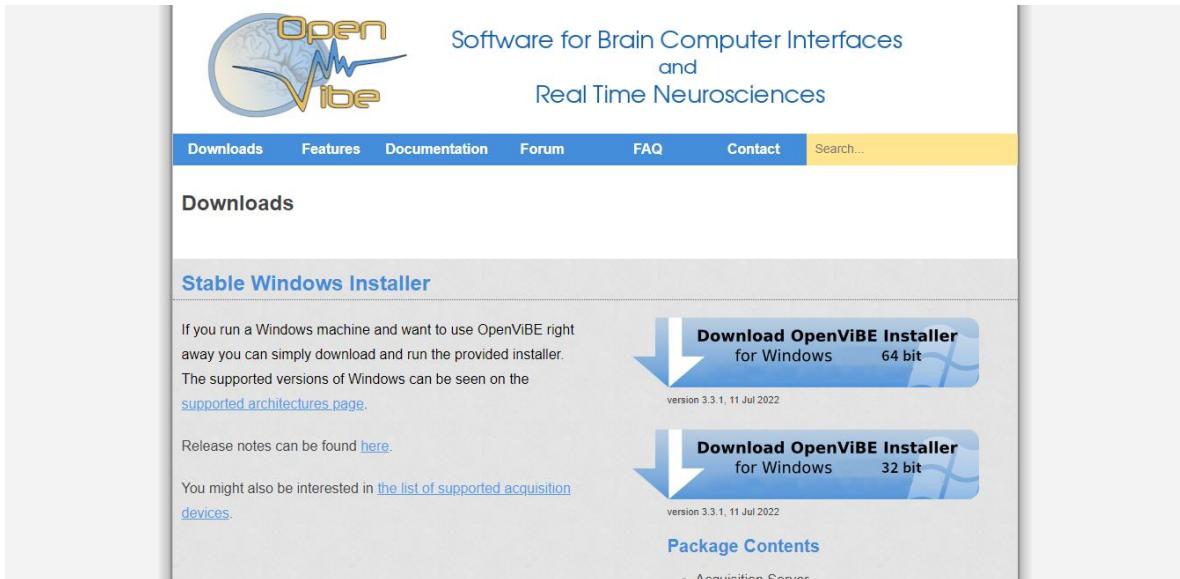


Figura 14.1.15. Sitio web de OpenViBE.

La versión que CymatiCorp recomienda para el trabajo presente es usar de la versión 2.2.0 en adelante. En el caso personal, se usará la versión 3.3.1 de 64 bits. De cumplir con este requisito, lo siguiente es ejecutar los programas “OpenViBE (Acquisition Server)” y “OpenViBE (Designer)”. La instalación no genera un acceso directo a estos programas, por lo que habrá que abrirlos en su mismo directorio o usando atajos que apunten a este directorio, según afirma CymatiCorp como algo necesario.

En OpenViBE (Acquisition Server) deberá seleccionarse “Generic Raw Telnet Reader” como Driver. El puerto de conexión (Connection port) es un puerto local al cual se conectará OpenViBE Designer. Este se deja por facilidad en 1024. Así también, en “Sample count per sent block” se dejará en 4. La configuración anterior puede observarse en la figura 14.1.16.

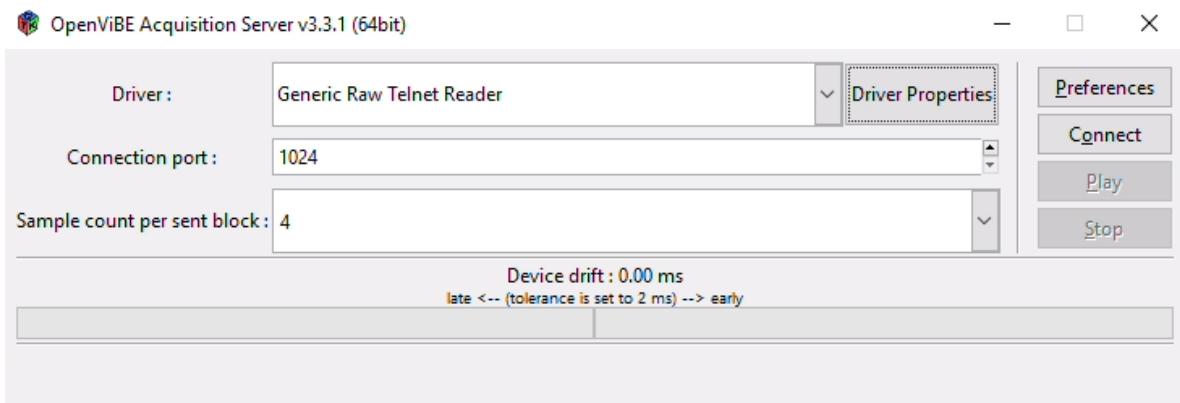


Figura 14.1.16. Configuración general de OpenViBE Acquisition Server.

Lo siguiente es modificar las opciones de propiedades del driver del botón “Driver Properties”. La configuración aparece en la figura 14.1.17 pero también se enumera a continuación la utilizada:

- **Number of channels:** 14.
- **Sampling frequency:** 256.
- **Telnet host name:** localhost.
- **Telnet host port:** 5151.
- **Endianess:** Big Endian.
- **Sample type:** 32 bits float.
- **Skip at start (bytes):** 0.
- **Skip header (bytes):** 0.
- **Skip footer (bytes):** 0.

Después de haber hecho estas configuraciones, se dará click en el botón de “Apply” y en OpenViBE Acquisition Server podrá dársele click en el botón “Connect” con CyKit.py ya ejecutándose para este punto del avance experimental. La conexión exitosa debe aparecer tanto en CyKit como en OpenViBE Acquisition Server como se muestra en la figura 14.1.18.

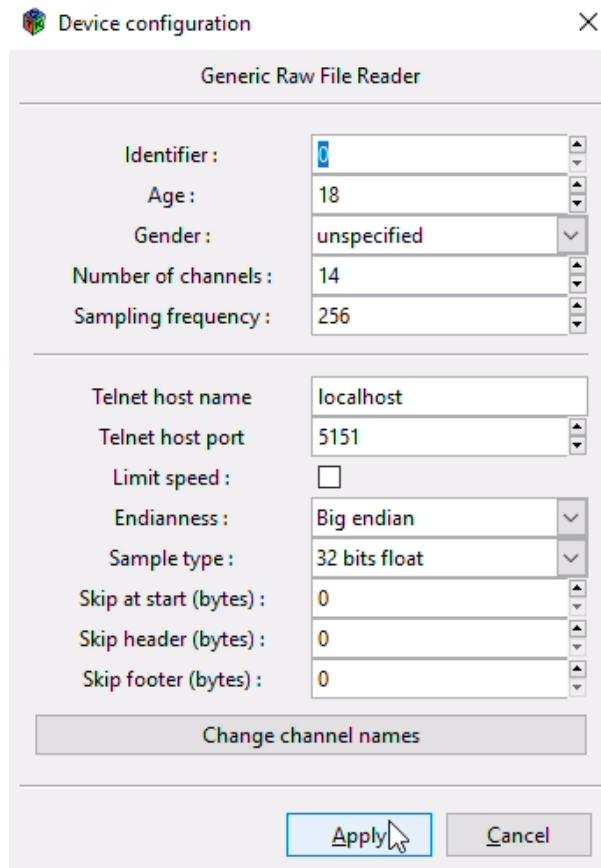


Figura 14.1.17. Configuración de propiedades de OpenViBE Acquisition Server.

Figura 14.1.18. Conexión exitosa entre CyKit y OpenViBE Acquisition Server.

Lo siguiente que debe hacerse es, ahora utilizar OpenViBE Designer, con el que se creará un escenario a través de arrastrar y soltar funciones al lienzo que tiene el diseñador. Para crear el escenario que marca CymatiCorp, será necesario seguir las instrucciones señaladas a continuación:

- 1)Arrastrar “Acquisition Client” que se puede encontrar dentro del directorio “Acquisition and network IO” en la barra lateral derecha.
 - 2)Arrastrar “Signal Display” y “Matrix Display” que se pueden encontrar dentro del directorio “Visualization” → “Basic” en la barra lateral derecha.
 - 3)Posicionarse sobre la flecha rosa de “Acquisition Client” y arrastrarla a la flecha verde de “Matrix Display”.
 - 4)Posicionarse sobre la flecha rosa de “Acquisition Client” y arrastrarla a la flecha rosa de “Signal Display”.
 - 5)Dar doble click sobre “Acquisition Client” y reemplazar “\${AcquisitionServer_HostName}” con “localhost”. Así también, asegurarse que “Acquisition server port” tenga un valor de “1024” si es que así se calibró en OpenViBE Acquisition Server.
 - 6)Finalmente aplicar los cambios.

Todo este conjunto de instrucciones puede encontrarse en el tutorial dado por CymatiCorp, liga dada previamente. Asimismo, el diagrama que se realizó con las anteriores instrucciones se vería reflejado de una forma similar a la ilustrada en la figura 14.1.19.

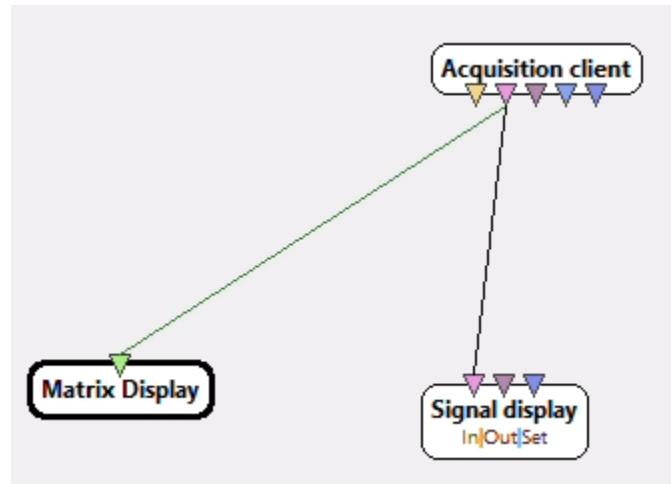


Figura 14.1.19. Ilustración del escenario creado en OpenViBE Designer.

Después de lo anterior, sólo queda darle al botón de ejecutar escenario y si todos los pasos se realizaron correctamente, un electroencefalograma y una tabla de valores deberían aparecer en pantalla, actualizando el estado de recepción de datos de la diadema; lo cual demuestra el funcionamiento correcto de la transmisión de datos. Esto se ilustra en las figuras 14.1.20 y 14.1.21.

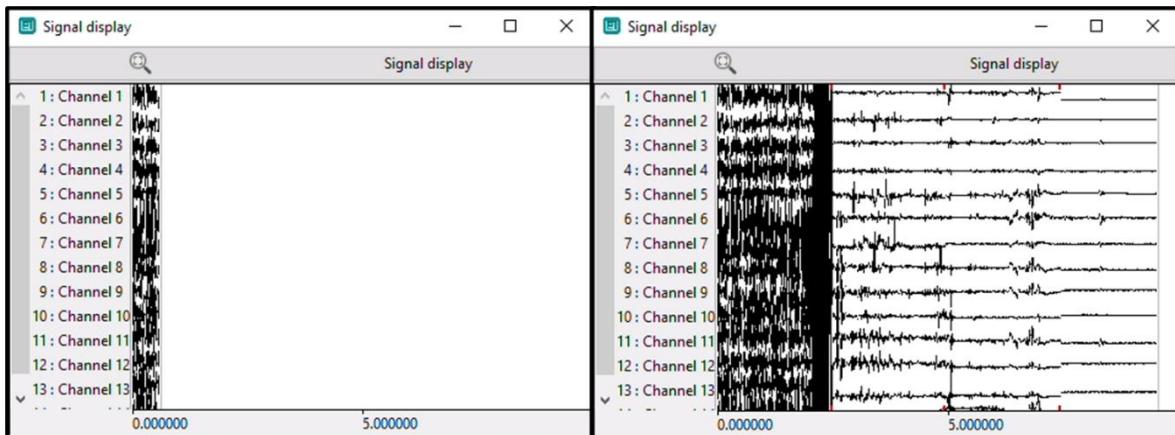


Figura 14.1.20. Electroencefalograma de OpenViBE Designer en funcionamiento.

Channel	1	2	3	4
Channel 1	4157.69	4158.33	4155.26	4162.82
Channel 2	4156.92	4159.10	4157.69	4160.00
Channel 3	4159.62	4162.44	4159.10	4157.69
Channel 4	4150.13	4157.69	4155.90	4159.23
Channel 5	4163.46	4152.69	4158.21	4153.08
Channel 6	4160.90	4159.74	4154.87	4157.05
Channel 7	4152.18	4156.79	4151.92	4160.13
Channel 8	4157.56	4158.59	4159.49	4153.85
Channel 9	4162.69	4162.05	4162.44	4159.36
Channel 10	4166.15	4163.97	4160.77	4160.38
Channel 11	4161.54	4159.62	4158.33	4161.15
Channel 12	4164.74	4166.28	4161.79	4163.97
Channel 13	4217.82	4214.49	4212.69	4211.54
Channel 14	4923.97	4922.95	4922.31	4923.85

Channel	1	2	3	4
Channel 1	4171.41	4171.41	4143.59	4140.13
Channel 2	4158.46	4158.46	4137.69	4129.36
Channel 3	4142.05	4142.05	4159.23	4149.49
Channel 4	4151.79	4151.79	4162.18	4153.46
Channel 5	4194.49	4194.49	4197.95	4193.72
Channel 6	4156.67	4156.67	4157.31	4155.00
Channel 7	4163.33	4163.33	4161.41	4161.54
Channel 8	4153.08	4153.08	4158.72	4161.28
Channel 9	4161.03	4161.03	4162.56	4158.72
Channel 10	4154.36	4154.36	4162.56	4161.15
Channel 11	4155.51	4155.51	4151.28	4153.33
Channel 12	4171.41	4171.41	4158.72	4163.21
Channel 13	4228.08	4228.08	4222.82	4218.08
Channel 14	4452.69	4452.69	4454.87	4461.54

Figura 14.1.21. Tabla de datos de OpenViBE Designer en funcionamiento correcto.

Hasta aquí llega la explicación de CymatiCorp para la transmisión de datos. Sin embargo, personalmente todavía no se sentía con mucha facilidad de hacer machine learning de esta forma de recopilar datos, por lo que se tuvieron que hacer algunos añadidos personales al aporte hecho por CymatiCorp.

Estos añadidos fueron dos. El primero está en el escenario realizado en OpenViBE Designer, pues al escenario se le agregará un proceso llamado “CSV File Writer” el cual puede encontrarse dentro del directorio “File reading and writing” → “CSV” en la barra lateral derecha. Después habrá que posicionarse sobre la flecha rosa de “Acquisition Client” y arrastrarla a la flecha rosa de “CSV File Writer”. Finalmente dando doble click sobre este último habrá que modificar el campo “Filename” al nombre y directorio de gusto propio. Así también debe marcarse como verdadera la casilla correspondiente a “Append data”. La figura 14.1.22 muestra el escenario modificado y la figura 14.1.23 muestra las configuraciones del proceso de “CSV File Writer”.

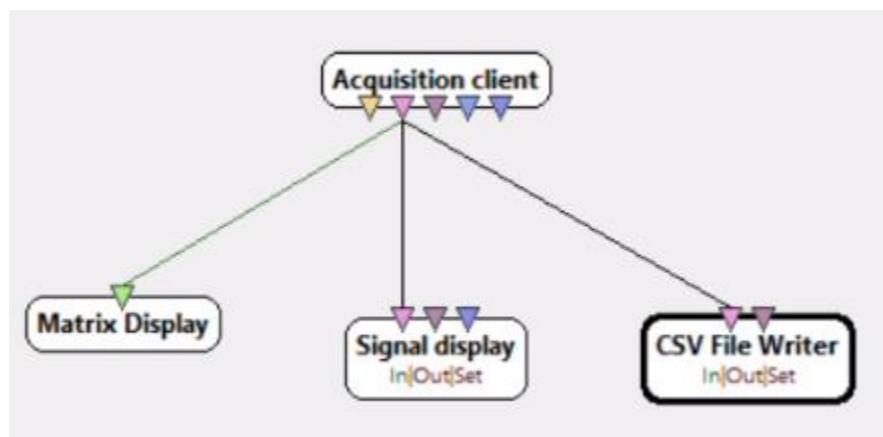


Figura 14.1.22. Escenario de OpenViBE Designer modificado personalmente.

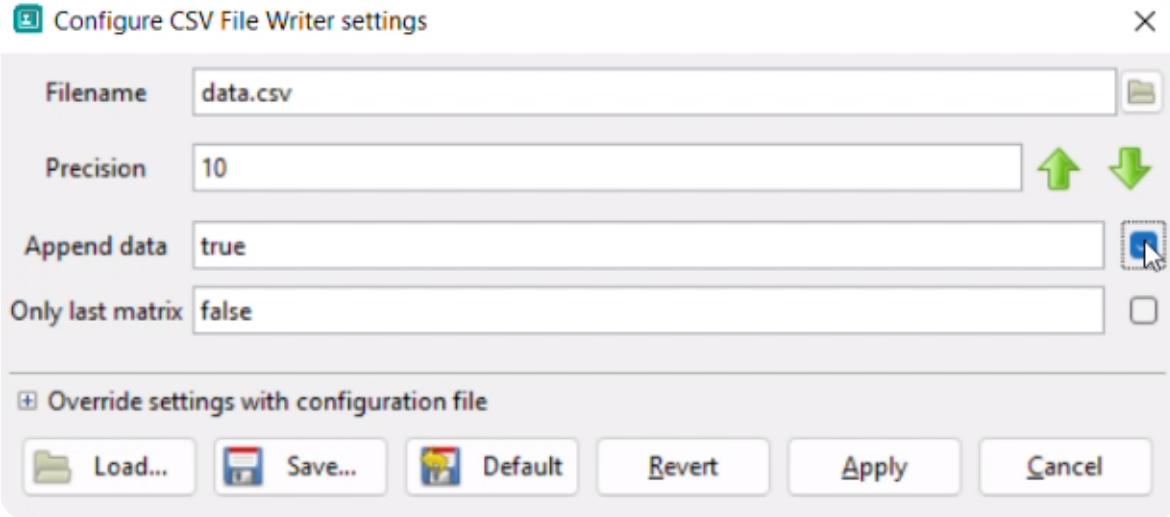


Figura 14.1.23. Configuraciones para “CSV File Writer”.

Esto finalmente permite una salida de datos mejor tratables para el machine learning realizado en esta investigación. Puede observarse en la figura 14.1.24 la tabla en formato CSV que produjo este cambio, con la que ya puede hacer más fácil la lectura de datos para los modelos de redes neuronales que se usarán en la presente investigación.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	Time:256Hz	Epoch	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6	Channel 7	Channel 8	Channel 9	Channel 10	Channel 11	Channel 12	Channel 13	Channel 14
2	0	0	4164.74365	4159.35889	4160.38477	4153.58984	4189.35889	4155.76904	4149.23096	4158.07715	4161.79492	4166.28223	4162.30762	4165.89746	4218.3335	4636.02588
3	0.00390625	0	4150.89746	4161.28223	4155.64111	4159.10254	4207.30762	4158.97436	4157.69238	4162.17969	4165.89746	4154.74365	4153.58984	4209.4873	4618.58984	
4	0.0078125	0	4160.76904	4158.71777	4162.69238	4158.07715	4192.94873	4151.6665	4151.15381	4160.12588	4164.74365	4163.84619	4166.53857	4168.84619	4215.12842	4635.5127
5	0.01171875	0	4154.61523	4161.28223	4154.74365	4158.58984	4206.53857	4160.5127	4159.4873	4153.97412	4160.87946	4162.56396	4160	4161.15381	4213.07715	4621.02588
6	0.015625	1	4161.53857	4158.71777	4158.84619	4150.5127	4157.94873	4153.58984	4162.94873	4161.15381	4166.53857	4161.28223	4163.46143	4214.10254	4630.25635	
7	0.01953125	1	4153.20508	4161.02588	4153.97412	4157.17969	4206.15381	4162.43604	4156.79492	4158.71777	4161.79492	4166.41016	4154.23096	4161.6665	4210.64111	4623.20508
8	0.0234375	1	4158.20508	4159.10254	4152.94873	4156.15381	4201.79492	4156.53857	4151.53857	4159.87158	4162.94873	4166.02588	4162.56396	4169.74365	4216.6665	4629.10254
9	0.02734375	1	4162.43604	4160.5127	4156.02588	4159.74365	4200.89746	4159.74365	4158.07715	4160	4160.76904	4166.15381	4166.15381	4166.79492	4218.07715	4623.84619
10	0.03125	2	4161.53857	4162.30762	4161.53857	4160.76904	4198.58984	4156.41016	4156.15381	4156.92285	4160.76904	4164.87158	4161.79492	4162.56396	4214.35889	4620.89746
11	0.03515625	2	4159.10254	4158.97412	4160.64111	4156.5288	4195.38477	4152.02588	4152.43604	4157.94873	4164.4873	4163.3335	4161.6665	4164.87158	4213.97412	4628.97412
12	0.0390625	2	4152.56396	4159.74365	4153.58984	4157.43604	4200	4158.97412	4157.56396	4158.84619	4165.64111	4164.35889	4160	4162.30762	4214.74365	4626.20223
13	0.04296875	2	4157.56396	4161.6665	4158.07715	4159.35889	4189.35889	4156.53857	4152.2031	4167.43604	4166.15381	4216.79492	4163.46143	4214.10254	4638.84619	
14	0.046875	3	4150.25635	4163.20508	4154.87158	4160.89746	4203.84619	4159.97412	4157.56396	4154.61523	4161.41016	4165.76904	4156.79492	4162.69238	4211.02588	4622.82031
15	0.05078125	3	4161.41016	4156.28223	4160.25635	4158.97412	4190.38477	4152.17969	4152.56396	4160.64111	4162.43604	4164.87158	4164.74365	4172.94873	4218.58984	4633.20508
16	0.0546875	3	4153.58984	4160.5127	4156.92285	4163.20508	4208.46143	4158.3335	4159.4873	4152.69238	4160.38477	4164.4873	4154.35889	4155.12842	4211.41016	4612.56396
17	0.05859375	3	4163.71777	4160.25635	4165.12842	4157.82031	4192.82031	4148.58984	4150	4157.30762	4162.02588	4164.74365	4166.53857	4215.5127	4634.35889	
18	0.0625	4	4150	4163.07715	4154.10254	4156.15381	4210.38477	4155.38477	4156.53857	4152.30762	4161.02588	4164.61523	4157.69238	4162.43604	4212.69238	4613.07715
19	0.06640625	4	4159.10254	4159.10254	4155	4155.76904	4190.25635	4146.15381	4145.26355	4162.17969	4161.53857	4166.92285	4161.02588	4166.92285	4217.82031	4623.58984
20	0.0703125	4	4155.76904	4163.84619	4150.25635	4162.30762	4026.92285	4159.61523	4157.69238	4155.61121	4161.41016	4167.82031	4154.87158	4157.94873	4211.53857	4617.56396
21	0.07421875	4	4162.69238	4161.53857	4157.17969	4159.35889	4192.17969	4155.25635	4152.2031	4160.5127	4162.69238	4165.76904	4165.5127	4169.4873	4217.82031	4636.15381
22	0.078125	5	4150.25635	4160.12842	4153.84619	4158.58984	4198.07715	4159.4873	4154.4873	4159.35889	4162.43604	4164.23096	4156.6665	4159.74365	4213.71777	4616.02588
23	0.08203125	5	4155.76904	4156.53857	4157.43604	4156.84619	4188.35894	4155.5127	4151.69238	4167.30762	4162.02588	4164.74365	4166.53857	4216.02588	4622.69238	
24	0.0859375	5	4157.56396	4158.07715	4155.76904	4160.12842	4200.5127	4158.3335	4160.52588	4158.97412	4161.79492	4162.56396	4160.5127	4162.17969	4214.87158	4621.28223
25	0.08984375	5	4159.10254	4161.28223	4158.3335	4156.41016	4200.38477	4156.79492	4152.82031	4156.41016	4161.41016	4165.12842	4164.4873	4213.97412	4619.74365	
26	0.09375	6	4154.35884	4162.43604	4160.64111	4153.58984	4198.97412	4158.87158	4148.71777	4162.30762	4162.94873	4163.46143	4161.15381	4213.3335	4624.4873	
27	0.09765625	6	4151.41016	4158.71777	4156.53857	4203.20508	4155.5127	4153.3335	4156.28223	4158.84619	4165.38477	4155.87492	4156.79492	4211.41016	4616.79492	
28	0.1015625	6	4163.46143	4158.84619	4159.61523	4157.56396	4195.12842	4154.74365	4152.56396	4160.52588	4170	4163.71777	4167.56393	4215.38477	4627.30762	
29	0.10546875	6	4153.71777	4163.84619	4156.28223	4158.46143	4208.84619	4159.10254	4154.23096	4154.35889	4163.02508	4166.15381	4154.35889	4160.25635	4210.5127	4617.30762
30	0.109375	7	4157.43604	4158.46143	4161.92285	4156.15381	4191.53857	4148.46143	4148.46143	4160.25635	4162.56396	4162.05127	4158.71777	4214.4873	4628.58984	
31	0.11328125	7	4153.58984	4160	4152.82031	4158.3335	4204.87158	4159.02588	4148.71777	4151.53857	4158.20508	4163.97412	4153.71777	4154.10254	4209.23096	4610.76904
32	0.1171875	7	4165.5127	4161.53851	4161.53851	4158.87158	4160.12582	4148.3335	4161.41016	4164.23096	4166.28223	4162.30762	4168.3335	4212.69238	4632.82031	
33	0.12109375	7	4150.76904	4164.35889	4157.82031	4158.46143	4208.84619	4159.4873	4157.82031	4154.87158	4161.28223	4161.41016	4153.97412	4158.97412	4208.84619	4617.30762
34	0.125	8	4161.6665	4159.74365	4158.97412	4156.79492	4189.35889	4153.71777	4155.12842	4160.25635	4158.46143	4163.3335	4166.53857	4168.46143	4221.41016	4634.61523
35	0.12890625	8	4157.43604	4163.07715	4151.53851	4158.58984	4203.84619	4164.10254	4159.61523	4156.53857	4157.69238	4166.6665	4160.5127	4215.89746	4615.38477	

Figura 14.1.24. Archivo CSV producido tras una ejecución de transmisión de datos desde la diadema.

Sin embargo, aún queda un problema más a resolver y es aquí donde está el segundo cambio notable; pues este archivo CSV se genera sólo hasta el final de la ejecución, es decir, cuando se detiene el proceso. Pero en una investigación y experimentación personal, se encontró una forma

de simular una lectura a tiempo real para poder realizar el análisis a tiempo real de los datos transmitidos desde la diadema al equipo, casi en vivo.

Esta forma de simulación se deberá hacer primeramente activando el ciclado del procedimiento. Esto se hace pulsando el botón de ciclado que está cerca del botón de ejecución, como se puede ver en la figura 14.1.25.



Figura 14.1.25. Botón para ciclar la ejecución.

Ahora, al haber activado esta opción, cada vez que se pulse el botón de detener el proceso, sólo se reiniciará el procedimiento, por lo que de esta manera puede simularse la transmisión en vivo de los datos.

Por último, para poder hacer este proceso automático se proponen dos archivos de Python que faciliten este trabajo. Estos archivos de Python requerirán de la librería “PyAutoGUI”, la cual puede instalarse con el comando de la figura 14.1.26

```
1 pip install pyautogui
```

Figura 14.1.26. Comando de instalación de la librería PyAutoGUI.

El propósito de estos archivos es que se pulse la detención cíclica del escenario periódicamente, lo cual hará la librería previamente instalada, pues la anterior permite controlar el teclado y el mouse de forma automática con código de Python.

Primeramente hay que encontrar las coordenadas del botón en la pantalla. Para ello se usaría el archivo Python de la figura 14.1.27.

```
1 import pyautogui
2 import time
3
4 # Aplicando delay para tener tiempo de acomodar el mouse sobre el botón
5 time.sleep(10)
6
7 # Obteniendo la posición del mouse
8 print(pyautogui.position())
```

Figura 14.1.27. Detección de la posición del mouse con la librería PyAutoGUI.

Ahora, cuando se tengan estos valores pueden guardarse en variables “x”, “y” que se pueden usar en el siguiente código, pues ahora, cuando todo el procedimiento de obtención de datos esté en ejecución: la diadema conectada, Emotiv en ejecución y OpenViBE transmitiendo datos, es momento de ejecutar el código siguiente para que la detención cíclica lo haga automáticamente el código de la figura 14.1.28.

```
- □ ×

1 import pyautogui
2 import time
3
4 x = 0
5 y = 0
6
7 # Aplicando delay al código para correr el escenario y automáticamente se
     mueva a la posición indicada
8 time.sleep(10)
9 pyautogui.moveTo(x, y)
10
11 # El proceso se mantiene en ejecución mientras el mouse esté en su posición
12 while(pyautogui.position() == (x, y)):
13     pyautogui.click()
14     time.sleep(1) # El CSV se actualiza cada segundo
```

Figura 14.1.28. Ejecución cíclica del mouse con la librería PyAutoGUI.

Una combinación de ambos en un único código sería la expuesta en la figura 14.1.29.

```
- □ ×

1 import pyautogui
2 import time
3
4 # Aplicando delay para tener tiempo de acomodar el mouse sobre el botón
5 time.sleep(10)
6
7 x, y = pyautogui.position()
8
9 # Aplicando delay al código para correr el escenario y automáticamente se
     mueva a la posición indicada
10 time.sleep(10)
11 pyautogui.moveTo(x, y)
12
13 # El proceso se mantiene en ejecución mientras el mouse esté en su posición
14 while(pyautogui.position() == (x, y)):
15     pyautogui.click()
16     time.sleep(1) # El CSV se actualiza cada segundo
```

Figura 14.1.29. Combinación de los códigos 14.1.27 y 14.1.28.

Y así, al ejecutar este código, se podrá actualizar el archivo CSV cada segundo, o según cuánto quiera personalizarse en la línea 16. Con esto ya podría realizarse machine learning con los datos de forma más efectiva para la presente investigación.

Este mismo procedimiento se documentó a su vez de forma audiovisual y se presenta en el siguiente enlace:

<https://youtu.be/MuBZdcLJXbk>

14.2. Anexo 2: Código del archivo de Python: functions.py.

El archivo de Python functions.py está constituido de las distintas funciones que a lo largo del procedimiento de investigación se realizaron y se compilaron en este documento. El archivo en su totalidad es el siguiente:

```
 1 #%% Libraries
 2 import math
 3 import cmath
 4 import os
 5 import time
 6 import random
 7 import pyautogui
 8 import pandas
 9 import numpy as np
10 import matplotlib.pyplot as plt
11 from sklearn.model_selection import train_test_split
12 from sklearn.svm import SVC
13 from sklearn.metrics import classification_report
14 from sklearn.metrics import precision_score
15 from sklearn.metrics import confusion_matrix
16 from sklearn.preprocessing import StandardScaler
17 from sklearn.ensemble import RandomForestClassifier
18 from sklearn.naive_bayes import GaussianNB
19 from scipy.stats import zscore
20 import sklearn.exceptions
21 import warnings
22
23 warnings.filterwarnings("ignore",
   category=sklearn.exceptions.UndefinedMetricWarning)
24
25 #%% Detect the position of the mouse
26 def MousePosition(secs = 3):
27     # Sleeping the script so we have time to set the mouse on its position
28     time.sleep(secs)
29
30     # Getting the mouse position
31     print(pyautogui.position())
32     return
33
34 #%% Run a live stream to update the dataset while streaming
35 def RunUpdateStream(xStop = 319, yStop = 60, xWindow = 319, yWindow = 8,
   timerRefresher = 1):
36     # Sleeping the script so we have time to run the schema and then
   automatically moves to the indicated position
37     time.sleep(10)
38     pyautogui.moveTo(xWindow, yWindow)
39
40     # The process will be okay while the mouse keeps on the indicated
   positions
41     while((pyautogui.position() == (xStop, yStop)) or (pyautogui.position() ==
   (xWindow, yWindow))):
42         # Clicking to move the EEG
43         pyautogui.moveTo(xWindow, yWindow)
44         pyautogui.click()
45         time.sleep(timerRefresher/2)
46
```

```
47      # Clicking to reload the CSV
48      pyautogui.moveTo(xStop, yStop)
49      pyautogui.click()
50      time.sleep(timerRefresher/2)
51
52  return
53
54 #%% Creating the dataset with the input data and the expected output
55 def createDataset(xMaximize, yMaximize, xPlay, yPlay, xWindow, yWindow, xStop,
56   yStop, profileName, expectedResult, rangeTime):
57     # Check if profile path exists
58     if(not(os.path.exists('profiles/' + str(profileName)))):
59       os.makedirs('profiles/' + str(profileName))
60     time.sleep(0.2)
61
62     # Maximizing OpenViBE Designer
63     pyautogui.moveTo(xMaximize, yMaximize)
64     pyautogui.click()
65     time.sleep(1)
66
67     # Playing OpenViBE Designer
68     pyautogui.moveTo(xPlay, yPlay)
69     pyautogui.click()
70     time.sleep(rangeTime)
71
72     # Clicking over OpenViBE Designer
73     pyautogui.moveTo(xWindow, yWindow)
74     pyautogui.click()
75     time.sleep(0.5)
76
77     # Stopping OpenViBE Designer
78     pyautogui.moveTo(xStop, yStop)
79     pyautogui.click()
80     time.sleep(0.5)
81
82     # Loading data.csv
83     dataframe = pandas.read_csv('data.csv')
84
85     # Reworking the dataframe
86     dataframe = dataframe.drop(labels='Time:256Hz', axis=1)
87     dataframe = dataframe.drop(labels='Epoch', axis=1)
88     dataframe = dataframe.drop(labels='Event Id', axis=1)
89     dataframe = dataframe.drop(labels='Event Date', axis=1)
90     dataframe = dataframe.drop(labels='Event Duration', axis=1)
91
92     # Add a column with the expected output
93     dataframe.insert(14, "Expected Output", [expectedResult]*len(dataframe))
94
95     # Append (or create) the dataset of this selected profile
96     if(not(os.path.isfile('profiles/' + str(profileName) + '/dataset.csv'))):
97       file = open('profiles/' + str(profileName) + '/dataset.csv', "x")
```

```

97         file.close()
98         dataframe.to_csv('profiles/' + str(profileName) + '/dataset.csv',
99             mode='a', index=False, header=True)
100        else:
101            dataframe.to_csv('profiles/' + str(profileName) + '/dataset.csv',
102                mode='a', index=False, header=False)
103
104    # Remove the data.csv raw file
105    os.remove('data.csv')
106
107 #%% First AI model
108 def SVM(profileName):
109     # Condition if profile or dataset doesn't exists
110     if(not(os.path.isfile('fourierEMF/' + str(profileName) +
111         '/dataset.csv'))):
112         print('User not found or dataset not created yet')
113         return
114
115     # Support Vector Machine process
116     df = pandas.read_csv('fourierEMF/' + str(profileName) + '/dataset.csv')
117
118     X = df.drop(['Expected Output'], axis = 1)
119     Y = df['Expected Output']
120
121     xtrain, xtest, ytrain, ytest = train_test_split(X, Y, test_size = 0.20)
122
123     classifier = SVC(kernel = 'linear')
124     classifier.fit(xtrain, ytrain)
125
126     ypred = classifier.predict(xtest)
127     print(ypred)
128
129     print(classification_report(ytest, ypred))
130
131     matriz = confusion_matrix(ytest, ypred)
132     print('Matriz de Confusión')
133     print(matriz)
134
135     precision = precision_score(ytest, ypred, pos_label="a", average=None)
136     print('Precisión del modelo')
137     print(precision)
138
139     return precision
140
141 #%% Second AI model
142 def NaiveBayes(profileName):
143     # Condition if profile or dataset doesn't exists
144     if(not(os.path.isfile('fourierEMF/' + str(profileName) +
145         '/dataset.csv'))):
146         print('User not found or dataset not created yet')
147         return
148
149     # Naive Bayes process

```

```
148     df = pandas.read_csv('fourierEMF/' + str(profileName) + '/dataset.csv')
149
150     X = df.drop(['Expected Output'], axis = 1)
151     Y = df['Expected Output']
152
153     xtrain, xtest, ytrain, ytest = train_test_split(X, Y, test_size = 0.20)
154
155     classifier = GaussianNB()
156     classifier.fit(xtrain, ytrain)
157
158     ypred = classifier.predict(xtest)
159
160     matriz = confusion_matrix(ytest, ypred)
161     print('Matriz de Confusión')
162     print(matriz)
163
164     precision = precision_score(ytest, ypred, pos_label="a", average=None)
165     print('Precisión del modelo')
166     print(precision)
167
168     return precision
169
170 #%% Third AI model
171 def RandomForest(profileName):
172     # Condition if profile or dataset doesn't exists
173     if(not(os.path.isfile('fourierEMF/' + str(profileName) +
174         '/dataset.csv'))):
175         print('User not found or dataset not created yet')
176         return
177
178     # Random Forest process
179     df = pandas.read_csv('fourierEMF/' + str(profileName) + '/dataset.csv')
180
181     X = df.drop(['Expected Output'], axis = 1)
182     Y = df['Expected Output']
183
184     xtrain, xtest, ytrain, ytest = train_test_split(X, Y, test_size = 0.20)
185
186     sc = StandardScaler()
187     xtrain = sc.fit_transform(xtrain)
188     xtest = sc.fit_transform(xtest)
189
190     classifier = RandomForestClassifier(n_estimators = 4)
191     classifier.fit(xtrain, ytrain)
192
193     ypred = classifier.predict(xtest)
194
195     matriz = confusion_matrix(ytest, ypred)
196     print('Matriz de Confusión')
197     print(matriz)
198
199     precision = precision_score(ytest, ypred, pos_label="a", average=None)
```

```
199     print('Precisión del modelo')
200     print(precision)
201
202     return precision
203
204 #%% Getting every precision value
205 def runModels(users):
206     resultsFile = open("results.txt", 'w')
207     resultsFile.write("SVM\n\n")
208     print("SVM")
209
210     for i in users:
211         resultsFile.write(i + "\n")
212         print("Usuario: " + i)
213         for j in range(5):
214             print("Iteración: " + str(j + 1))
215             result = SVM(i)
216             resultsFile.write(str(result) + "\n")
217             resultsFile.close()
218             resultsFile = open("results.txt", 'a')
219             resultsFile.write("\n")
220
221     resultsFile.write("Naive Bayes\n\n")
222     print("Naive Bayes")
223
224     for i in users:
225         resultsFile.write(i + "\n")
226         print("Usuario: " + i)
227         for j in range(5):
228             print("Iteración: " + str(j + 1))
229             result = NaiveBayes(i)
230             resultsFile.write(str(result) + "\n")
231             resultsFile.close()
232             resultsFile = open("results.txt", 'a')
233             resultsFile.write("\n")
234
235     resultsFile.write("Random Forest\n\n")
236     print("Random Forest")
237
238     for i in users:
239         resultsFile.write(i + "\n")
240         print("Usuario: " + i)
241         for j in range(5):
242             print("Iteración: " + str(j + 1))
243             result = RandomForest(i)
244             resultsFile.write(str(result) + "\n")
245             resultsFile.close()
246             resultsFile = open("results.txt", 'a')
247             resultsFile.write("\n")
248
249     resultsFile.close()
```

```
250     return
251
252 #%% Z-Normalization
253 def NormZ(profileName):
254     for i in profileName:
255         # Condition if profile or dataset doesn't exists
256         if(not(os.path.isfile('normz/' + str(i) + '/dataset.csv'))):
257             print('User not found or dataset not created yet')
258             return
259
260     # Z-Normalization
261     df = pandas.read_csv('normz/' + str(i) + '/dataset.csv')
262
263     df['Channel 1'] = zscore(df['Channel 1'])
264     df['Channel 2'] = zscore(df['Channel 2'])
265     df['Channel 3'] = zscore(df['Channel 3'])
266     df['Channel 4'] = zscore(df['Channel 4'])
267     df['Channel 5'] = zscore(df['Channel 5'])
268     df['Channel 6'] = zscore(df['Channel 6'])
269     df['Channel 7'] = zscore(df['Channel 7'])
270     df['Channel 8'] = zscore(df['Channel 8'])
271     df['Channel 9'] = zscore(df['Channel 9'])
272     df['Channel 10'] = zscore(df['Channel 10'])
273     df['Channel 11'] = zscore(df['Channel 11'])
274     df['Channel 12'] = zscore(df['Channel 12'])
275     df['Channel 13'] = zscore(df['Channel 13'])
276     df['Channel 14'] = zscore(df['Channel 14'])
277
278     df.to_csv('normz/' + str(i) + '/dataset.csv', index = False)
279
280     return
281
282 #%% Fourier Transformation
283 def Fourier(profileName):
284     for i in profileName:
285         # Condition if profile or dataset doesn't exists
286         if(not(os.path.isfile('fourier/' + str(i) + '/dataset.csv'))):
287             print('User not found or dataset not created yet')
288             return
289
290     # Fourier Transformation
291     df = pandas.read_csv('fourier/' + str(i) + '/dataset.csv')
292
293     channel1 = np.array(df['Channel 1'])
294     channel2 = np.array(df['Channel 2'])
295     channel3 = np.array(df['Channel 3'])
296     channel4 = np.array(df['Channel 4'])
297     channel5 = np.array(df['Channel 5'])
298     channel6 = np.array(df['Channel 6'])
299     channel7 = np.array(df['Channel 7'])
300     channel8 = np.array(df['Channel 8'])
```

```
301     channel9 = np.array(df['Channel 9'])
302     channel10 = np.array(df['Channel 10'])
303     channel11 = np.array(df['Channel 11'])
304     channel12 = np.array(df['Channel 12'])
305     channel13 = np.array(df['Channel 13'])
306     channel14 = np.array(df['Channel 14'])
307
308     transform1 = np.fft.fft(channel1)
309     transform2 = np.fft.fft(channel2)
310     transform3 = np.fft.fft(channel3)
311     transform4 = np.fft.fft(channel4)
312     transform5 = np.fft.fft(channel5)
313     transform6 = np.fft.fft(channel6)
314     transform7 = np.fft.fft(channel7)
315     transform8 = np.fft.fft(channel8)
316     transform9 = np.fft.fft(channel9)
317     transform10 = np.fft.fft(channel10)
318     transform11 = np.fft.fft(channel11)
319     transform12 = np.fft.fft(channel12)
320     transform13 = np.fft.fft(channel13)
321     transform14 = np.fft.fft(channel14)
322
323     df['Channel 1'] = transform1
324     df['Channel 2'] = transform2
325     df['Channel 3'] = transform3
326     df['Channel 4'] = transform4
327     df['Channel 5'] = transform5
328     df['Channel 6'] = transform6
329     df['Channel 7'] = transform7
330     df['Channel 8'] = transform8
331     df['Channel 9'] = transform9
332     df['Channel 10'] = transform10
333     df['Channel 11'] = transform11
334     df['Channel 12'] = transform12
335     df['Channel 13'] = transform13
336     df['Channel 14'] = transform14
337
338     df.to_csv('fourier/' + str(i) + '/dataset.csv', index = False)
339
340     return
341
342 #%% Fourier Transformation (approach based on magnitude and phase)
343 def FourierEMF(profileName):
344     for i in profileName:
345         # Condition if profile or dataset doesn't exists
346         if(not(os.path.isfile('fourierEMF/' + str(i) + '/dataset.csv'))):
347             print('User not found or dataset not created yet')
348             return
349
350     # Application of the approach
351     print(i)
```

```

352
353     df = pandas.read_csv('fourierEMF/' + str(i) + '/dataset.csv')
354     channel1m = []
355     channel1f = []
356     channel2m = []
357     channel2f = []
358     channel3m = []
359     channel3f = []
360     channel4m = []
361     channel4f = []
362     channel5m = []
363     channel5f = []
364     channel6m = []
365     channel6f = []
366     channel7m = []
367     channel7f = []
368     channel8m = []
369     channel8f = []
370     channel9m = []
371     channel9f = []
372     channel10m = []
373     channel10f = []
374     channel11m = []
375     channel11f = []
376     channel12m = []
377     channel12f = []
378     channel13m = []
379     channel13f = []
380     channel14m = []
381     channel14f = []
382     for j in range(len(df['Expected Output'])):
383         channel1 = complex(df['Channel 1'][j])
384         channel2 = complex(df['Channel 2'][j])
385         channel3 = complex(df['Channel 3'][j])
386         channel4 = complex(df['Channel 4'][j])
387         channel5 = complex(df['Channel 5'][j])
388         channel6 = complex(df['Channel 6'][j])
389         channel7 = complex(df['Channel 7'][j])
390         channel8 = complex(df['Channel 8'][j])
391         channel9 = complex(df['Channel 9'][j])
392         channel10 = complex(df['Channel 10'][j])
393         channel11 = complex(df['Channel 11'][j])
394         channel12 = complex(df['Channel 12'][j])
395         channel13 = complex(df['Channel 13'][j])
396         channel14 = complex(df['Channel 14'][j])
397
398         channel1m.append(math.sqrt((channel1.real ** 2) + (channel1.imag
399             ** 2)))
400         channel2m.append(math.sqrt((channel2.real ** 2) + (channel2.imag
401             ** 2)))
402         channel3m.append(math.sqrt((channel3.real ** 2) + (channel3.imag
403             ** 2)))
404         channel4m.append(math.sqrt((channel4.real ** 2) + (channel4.imag
405             ** 2)))
406         channel5m.append(math.sqrt((channel5.real ** 2) + (channel5.imag
407             ** 2)))

```

```

403      channel6m.append(math.sqrt((channel6.real ** 2) + (channel6.imag
404          ** 2)))
405      channel7m.append(math.sqrt((channel7.real ** 2) + (channel7.imag
406          ** 2)))
407      channel8m.append(math.sqrt((channel8.real ** 2) + (channel8.imag
408          ** 2)))
409      channel9m.append(math.sqrt((channel9.real ** 2) + (channel9.imag
410          ** 2)))
411      channel10m.append(math.sqrt((channel10.real ** 2) +
412          (channel10.imag ** 2)))
413      channel11m.append(math.sqrt((channel11.real ** 2) +
414          (channel11.imag ** 2)))
415      channel12m.append(math.sqrt((channel12.real ** 2) +
416          (channel12.imag ** 2)))
417      channel13m.append(math.sqrt((channel13.real ** 2) +
418          (channel13.imag ** 2)))
419      channel14m.append(math.sqrt((channel14.real ** 2) +
420          (channel14.imag ** 2)))

421      channel1f.append(cmath.phase(channel1))
422      channel2f.append(cmath.phase(channel2))
423      channel3f.append(cmath.phase(channel3))
424      channel4f.append(cmath.phase(channel4))
425      channel5f.append(cmath.phase(channel5))
426      channel6f.append(cmath.phase(channel6))
427      channel7f.append(cmath.phase(channel7))
428      channel8f.append(cmath.phase(channel8))
429      channel9f.append(cmath.phase(channel9))
430      channel10f.append(cmath.phase(channel10))
431      channel11f.append(cmath.phase(channel11))
432      channel12f.append(cmath.phase(channel12))
433      channel13f.append(cmath.phase(channel13))
434      channel14f.append(cmath.phase(channel14))

435      df['Channel 1 M'] = channel1m
436      df['Channel 1 F'] = channel1f
437      df['Channel 2 M'] = channel2m
438      df['Channel 2 F'] = channel2f
439      df['Channel 3 M'] = channel3m
440      df['Channel 3 F'] = channel3f
441      df['Channel 4 M'] = channel4m
442      df['Channel 4 F'] = channel4f
443      df['Channel 5 M'] = channel5m
444      df['Channel 5 F'] = channel5f
445      df['Channel 6 M'] = channel6m
446      df['Channel 6 F'] = channel6f
447      df['Channel 7 M'] = channel7m
448      df['Channel 7 F'] = channel7f
449      df['Channel 8 M'] = channel8m
450      df['Channel 8 F'] = channel8f
451      df['Channel 9 M'] = channel9m
452      df['Channel 9 F'] = channel9f
453      df['Channel 10 M'] = channel10m

```

```
454     df['Channel 14 M'] = channel14m
455     df['Channel 14 F'] = channel14f
456
457     df = df.drop(['Channel 1'], axis = 1)
458     df = df.drop(['Channel 2'], axis = 1)
459     df = df.drop(['Channel 3'], axis = 1)
460     df = df.drop(['Channel 4'], axis = 1)
461     df = df.drop(['Channel 5'], axis = 1)
462     df = df.drop(['Channel 6'], axis = 1)
463     df = df.drop(['Channel 7'], axis = 1)
464     df = df.drop(['Channel 8'], axis = 1)
465     df = df.drop(['Channel 9'], axis = 1)
466     df = df.drop(['Channel 10'], axis = 1)
467     df = df.drop(['Channel 11'], axis = 1)
468     df = df.drop(['Channel 12'], axis = 1)
469     df = df.drop(['Channel 13'], axis = 1)
470     df = df.drop(['Channel 14'], axis = 1)
471
472     df.to_csv('fourierEMF/' + str(i) + '/dataset.csv', index = False)
473
474     return
```

Figura 14.2.1. Código del archivo functions.py utilizado en la experimentación.

14.3. Anexo 3: Pseudocódigo de la función RunUpdateStream() encontrada en el archivo de Python: functions.py.

El acercamiento para nombrar las variables y revisar el procedimiento a modo de boceto fue el siguiente:

Las variables a utilizar serán las siguientes:

- xStop: La posición en x del botón de Stop.
- yStop: La posición en y del botón de Stop.
- xWindow: La posición en x en otro punto de la pantalla (para evitar que el EEG se interponga).
- yWindow: La posición en y en otro punto de la pantalla (para evitar que el EEG se interponga).
- timerRefresher: El tiempo que reposa el código que ocurre entre click y click para refrescar el data.csv.

El procedimiento será el siguiente:

- 1) Se asignan xStop y yStop a los valores que corresponden.
- 2) Se asignan xWindow y yWindow a los valores que corresponden.
- 3) Mientras la posición del mouse sea (xStop, yStop) o (xWindow, yWindow):
 - a. El mouse hace click sobre la posición (xWindow, yWindow).
 - b. El sistema hace un sleep de:

$$\frac{\text{timerRefresher}}{2}$$

- c. El mouse hace click sobre la posición (xStop, yStop).
- d. El sistema hace un sleep de:

$$\frac{\text{timerRefresher}}{2}$$

14.4. Anexo 4: Pseudocódigo de la función createDataset() encontrada en el archivo de Python: functions.py.

El acercamiento para nombrar las variables y revisar el procedimiento a modo de boceto fue el siguiente:

Las variables a usar serán:

- xMaximize: La coordenada en x de donde se dará click para maximizar OpenViBE Designer.
- yMaximize: La coordenada en y de donde se dará click para maximizar OpenViBE Designer.
- xPlay: La coordenada en x de donde se dará click para darle play a OpenViBE Designer.
- yPlay: La coordenada en y de donde se dará click para darle play a OpenViBE Designer.
- xWindow: La coordenada en x de donde se da click para superponer a OpenViBE Designer.
- yWindow: La coordenada en y de donde se da click para superponer a OpenViBE Designer.
- xStop: La coordenada en x de donde se dará click para darle stop a OpenViBE Designer.
- yStop: La coordenada en y de donde se dará click para darle stop a OpenViBE Designer.
- profileName: El nombre de perfil que se usará.
- expectedResult: El resultado que se espera en la ejecución de la construcción del dataset.
- rangeTime: El tiempo en segundos que se empleará en la ejecución de la función.

El procedimiento será el siguiente:

- 1) Se checa si existe en la carpeta “profiles” una carpeta con nombre “profileName”. Si no existe, se crea esta carpeta y esta se asigna como workspace, si existe directamente se asigna como workspace.
- 2) La función maximiza la ventana de OpenViBE Designer dándole click a (xMaximize, yMaximize).
- 3) Posteriormente la función da click en play en la ejecución del Designer en (xPlay, yPlay).
- 4) La función hará un sleep durante rangeTime recopilando los datos.
- 5) Al terminar este sleep, por precaución se da click en (xWindow, yWindow) y posteriormente a (xStop, yStop).
- 6) Python cargará el “data.csv” que se genera en el mismo directorio donde se encuentra el script para añadirle una columna extra “Expected Result” donde se añadiría la variable “expectedResult”.
- 7) Este DataFrame se añade al archivo “dataset.csv” que se ubique en la carpeta del nombre de perfil dentro de “profiles”. Si el archivo no existe, se crea, si sí existe, sólo se le hace un append al archivo ya existente.
- 8) Se elimina “data.csv”.

14.5. Anexo 5: Acuerdo de privacidad entregado a los usuarios que colaboraron con el experimento.

En la página siguiente se adjunta el acuerdo de privacidad elaborado para asegurar a los usuarios que colaboraron con el experimento, que sus datos serían tratados éticamente según los fines de esta investigación y los permisos cedidos por ellos mismos.

ACUERDO DE PRIVACIDAD

1. Introducción

Este acuerdo establece los términos y condiciones en los cuales los participantes (en adelante "Usuarios") en el estudio experimental "Implementación de una Interfaz Cerebro – Computadora por medio de la Diadema Emotiv Epoc+ y el Programa Cykit Utilizando Modelos de Aprendizaje Supervisado. Tesina de Joel Alejandro Espinoza Sánchez para Obtener el Grado de Ingeniero en Computación Inteligente" (en adelante "Estudio") ceden el derecho de uso de los datos experimentales generados con su participación por los investigadores en el estudio (en adelante "Investigadores"). Al firmar este acuerdo, los Usuarios aceptan las condiciones establecidas en este documento.

2. Objeto del acuerdo

Los Usuarios ceden al equipo de investigación el derecho de usar los datos experimentales generados con su participación en el estudio. Los datos que se generarán para el estudio son generados a través de la diadema Emotiv Epoc+. La información utilizada será únicamente el comportamiento cerebral generado por los Usuarios que registre la diadema. Los datos podrán ser utilizados para la realización de investigaciones, publicaciones y presentaciones en eventos científicos.

3. Tratamiento de los datos

Los Investigadores se comprometen a proteger la privacidad de los Usuarios y a tratar los datos de manera confidencial. Los datos se almacenarán de manera segura y solo serán accesibles por el equipo de investigación. Los datos serán anonimizados antes de ser compartidos con terceros, y solo se compartirán con instituciones y organizaciones con fines científicos.

4. Derechos de los Usuarios

Los Usuarios tienen derecho a acceder, rectificar y suprimir sus datos en cualquier momento, siempre y cuando no interfiera con el desarrollo del Estudio. También tienen derecho a retirar su consentimiento

en cualquier momento, siempre y cuando no afecte a la validez de los datos ya recolectados.

5. Permisos cedidos por el Usuario

El Usuario cede los permisos señalados a continuación como "Acepto" y restringe aquellos indicados como "No Acepto" señalizados y elegidos por el mismo Usuario:

- a) Que se me incluya en el proceso experimental del Estudio proporcionando mis datos relacionados a la actividad cerebral, generados con la diadema Emotiv Epoc+.
Acepto: No Acepto: _____
- b) Que se utilice como dato personal en la redacción, publicación y presentación del Estudio y subsecuentes, mi edad.
Acepto: No Acepto: _____
- c) Que se utilice como dato personal en la redacción, publicación y presentación del Estudio y subsecuentes, mi nombre completo.
Acepto: _____ No Acepto: _____
- d) Que se puedan realizar grabaciones de audio y video incluyendo mi imagen únicamente como parte de la evidencia del proceso de realización del experimento del Estudio para el control de avances requerido por docentes y sinodales evaluadores del Estudio.
Acepto: _____ No Acepto: _____
- e) Que el material audiovisual generado con mi imagen sea usado en la redacción, publicación y presentación del Estudio y subsecuentes (No aplica si no se acepta el inciso C).
Acepto: _____ No Acepto: _____

6. Vigencia

Este acuerdo tendrá vigencia hasta la finalización del estudio, y será rescindido una vez que se hayan cumplido todos los objetivos del mismo.

7. Aceptación del acuerdo

La firma del presente acuerdo por parte del Usuario implica la aceptación de las condiciones establecidas en este documento.

Firma del Usuario

Fecha

Firma del Investigador

Fecha

14.6. Anexo 6: Protocolo de experimentación.

Se adjunta en el presente anexo el protocolo de experimentación de campo realizado para la toma de muestras de los usuarios de prueba. El protocolo fue realizado para facilitar la toma de la gran cantidad de muestras así como de estandarizar el entorno de hardware y software para todos los usuarios.

Se aclaran los roles “usuario” e “investigador” como el usuario quien se probará la diadema y de quien se recolectarán datos y el supervisor de la prueba el cual está encargado de la coordinación de la experimentación respectivamente

Previo a la experimentación

1. El investigador debe preparar el área de experimentación. Esto implica encontrar y preparar un espacio con poco ruido
2. El investigador debe preparar el equipo de experimentación. Esto implica preparar el equipo de cómputo propio:
 1. Tener el equipo encendido y con batería
 2. Tener abierto el programa de Spyder con el script de uso con el comando de creación del dataset preparado para ejecutarlo
 3. Tener abierto Emotiv Launcher
 4. Tener preparado el comando de ejecución de CyKit
 5. Tener abiertos OpenViBE Designer y OpenViBE Acquisition Server con las configuraciones del esquema apropiadas
3. El investigador debe preparar la diadema para la experimentación.
 1. Sacar la diadema de la caja
 2. Humedecer por aproximadamente cinco minutos las almohadillas en un bote con agua y después retirarlas para reposar del agua en una toalla
 3. Acomodar las almohadillas y los nodos eléctricos en cada uno de los receptores de la diadema
 4. Encender la diadema
 5. Conectar el USB inalámbrico al equipo
4. El investigador debe otorgarle al usuario el acuerdo de privacidad donde éste tiene que aceptar los términos para poder proseguir con el actual usuario
5. Si el usuario acepta los términos del acuerdo de privacidad, puede proseguirse con la experimentación

Durante la experimentación

1. El investigador acomodará la diadema en la cabeza del usuario de modo que Emotiv Launcher muestre un alto porcentaje de recepción
2. Una vez que el porcentaje sea alto, se realizan las conexiones de software:

1. Se ejecuta CyKit:

```
1 cd 'CyKit\Py3'  
2 python CyKIT.py 127.0.0.1 5151 6  
    openvibe+generic+nocounter+noheader+nobattery+ovdelay:100+float+ovsamples:004
```

2. Se conecta OpenViBE Acquisition Server
3. Al tener las conexiones realizadas, se ejecutará el comando en Python que crea el dataset:

```
1 createDataset(1211, 1048, 453, 77, 452, 15, 398, 77, 'perfil1', "Arbol", 10)
```

Se nombra la experimentación con el nombre del usuario seguido de un 1 y en este dataset se le pedirá al usuario:

1. Un árbol
 2. Un cuaderno
 3. Una computadora
 4. Un perro
 5. Evitar la concentración de algún pensamiento en específico
4. Al tener las conexiones realizadas, se ejecutará el comando en Python que crea el dataset:

```
1 createDataset(1211, 1048, 453, 77, 452, 15, 398, 77, 'perfil2', "MouseUp", 10)
```

Se nombra la experimentación con el nombre del usuario seguido de un 1 y en este dataset se le pedirá al usuario:

1. Mover el mouse hacia arriba
 2. Mover el mouse hacia abajo
 3. Mover el mouse hacia izquierda
 4. Mover el mouse hacia derecha
 5. Evitar la concentración de algún pensamiento en específico
5. Finalmente el investigador retira la diadema del usuario revisando que todas las almohadillas estén en la diadema o de lo contrario buscarlas previo a que el usuario se retire

6. Al verificar que el número de almohadillas sea el correcto, el investigador puede dejar ir al usuario

Después de la experimentación

Si existen más usuarios después del recientemente finalizado, se repite desde el paso 4 previo a la experimentación hasta el paso 6 durante la experimentación para un nuevo usuario

1. Al terminar la experimentación se cierra Spyder, posteriormente se desconecta y se apaga el Acquisition Server, luego se cierra el Designer y el Acquisition Server y finalmente CyKit
2. Después se desconecta la USB de la diadema y se termina de apagar el equipo de cómputo
3. La diadema se desarma eliminando las almohadillas y los nodos eléctricos dejando todo en su caja

Nota 1: Si se quiere grabar la pantalla de la computadora lo mejor es que se haga por usuario.

Nota 2: El comando de Python funciona con las coordenadas con la siguiente barra de inicio:



14.7. Anexo 7: Evidencias de la experimentación de campo.

La experimentación de campo consistió en conseguir datos a partir de personas voluntarias a ser usuarios de prueba dentro de este rubro, ofreciéndose a probarse la diadema y recopilar datos de su actividad cerebral para con ellos poner a prueba los modelos de machine learning mencionados.

Los participantes en el experimento cedieron los permisos indicados en la tabla 14.7.1 donde incluye también el número de participante al que se hace referencia en los resultados del experimento (cabe aclarar que como el investigador también aportó datos, éste es el participante número uno). Antes que nada personalmente he de agradecer a todos y cada uno de los participantes involucrados en la experimentación pues aportaron en una gran medida al desarrollo de esta investigación.

Tabla 14.7.1. Usuarios voluntarios en la experimentación de campo junto a los términos cedidos en el acuerdo.

#	Nombre	Uso del nombre	Grabación del usuario	Uso de las evidencias
2	José Luis Espinoza Sánchez	Sí	Sí	Sí
3	María Consuelo Sánchez Díaz	Sí	Sí	Sí
4	José Luis Espinoza González	Sí	Sí	Sí
5	Román Guadalupe de León Vázquez	Sí	Sí	Sí
6	Hiram Efraín Orocio García	Sí	Sí	Sí
7	Andrea Melissa Almeida Ortega	Sí	Sí	Sí
8	Óscar Alonso Flores Fernández	Sí	Sí	Sí
9	Edith Berenice Orocio García	Sí	Sí	Sí
10	Andrea de Santiago Legaspi	Sí	Sí	Sí
11	Montserrat Alejandra Ulloa Rivera	Sí	Sí	Sí
12	Juan Abraham Ortíz Salas	Sí	Sí	Sí
13	Valeria Macías Soto	Sí	Sí	Sí
14	Andrea Julieth Ordaz de Vierna	Sí	Sí	Sí
15	Gustavo Bejamín II Pedraza Morales	Sí	Sí	Sí
16	Alejandro Ramos Herrera	Sí	Sí	Sí
17	Miguel Ángel Meza de Luna	Sí	Sí	Sí
18	Fernando Ulloa	Sí	Sí	Sí
19	Andrés Eloy Escobedo Esparza	Sí	Sí	Sí
20	Jorge Luis Villalobos Araiza	Sí	Sí	Sí
21	Diego Romo Cruz	Sí	Sí	Sí
22	Raúl	Sí	Sí	Sí
23	Alejandro Padilla Díaz	Sí	Sí	Sí
24	Nancy Yissel Cuéllar Valdivia	Sí	Sí	Sí
25	Daniela del Carmen Fuentes Acata	Sí	Sí	Sí
26	Isabel Cristina Durón Esparza	Sí	Sí	Sí
27	Yolanda Muñoz Gaytán	Sí	Sí	Sí

Cuando se realizó la experimentación de campo se documentó todo en video de la misma forma en la que los datos obtenidos están guardados en archivos de valores separados por comas evidenciado en la figura 14.7.1.

joul1	13/02/2023 04:43 p. m.	Carpeta de archivos
joul2	13/02/2023 04:47 p. m.	Carpeta de archivos
pep1	13/02/2023 05:08 p. m.	Carpeta de archivos
pep2	13/02/2023 05:18 p. m.	Carpeta de archivos
mom1	13/02/2023 08:52 p. m.	Carpeta de archivos
mom2	13/02/2023 08:55 p. m.	Carpeta de archivos
dad1	13/02/2023 09:04 p. m.	Carpeta de archivos
dad2	13/02/2023 09:08 p. m.	Carpeta de archivos
Roman1	14/02/2023 08:09 a. m.	Carpeta de archivos
Roman2	14/02/2023 08:13 a. m.	Carpeta de archivos
Hiram1	14/02/2023 08:23 a. m.	Carpeta de archivos
Hiram2	14/02/2023 08:27 a. m.	Carpeta de archivos
Meli1	14/02/2023 08:34 a. m.	Carpeta de archivos
Meli2	14/02/2023 08:37 a. m.	Carpeta de archivos
Oscar1	14/02/2023 08:45 a. m.	Carpeta de archivos
Oscar2	14/02/2023 08:50 a. m.	Carpeta de archivos
Bere1	14/02/2023 11:13 a. m.	Carpeta de archivos
Bere2	14/02/2023 11:18 a. m.	Carpeta de archivos
Andy1	14/02/2023 11:52 a. m.	Carpeta de archivos
Andy2	14/02/2023 11:56 a. m.	Carpeta de archivos
Montse1	14/02/2023 12:06 p. m.	Carpeta de archivos
Montse2	14/02/2023 12:09 p. m.	Carpeta de archivos
Abraham1	14/02/2023 12:19 p. m.	Carpeta de archivos
Abraham2	14/02/2023 12:25 p. m.	Carpeta de archivos
Valeria1	14/02/2023 12:34 p. m.	Carpeta de archivos
Valeria2	14/02/2023 12:37 p. m.	Carpeta de archivos
Juliett1	14/02/2023 12:43 p. m.	Carpeta de archivos
Juliett2	14/02/2023 12:46 p. m.	Carpeta de archivos
Gus1	14/02/2023 01:39 p. m.	Carpeta de archivos
Gus2	14/02/2023 01:45 p. m.	Carpeta de archivos
Alex1	15/02/2023 10:29 a. m.	Carpeta de archivos
Alex2	15/02/2023 10:31 a. m.	Carpeta de archivos
Meza1	15/02/2023 10:38 a. m.	Carpeta de archivos
Meza2	15/02/2023 10:41 a. m.	Carpeta de archivos
Fer1	15/02/2023 10:49 a. m.	Carpeta de archivos
Fer2	15/02/2023 10:52 a. m.	Carpeta de archivos
Eloy1	15/02/2023 01:17 p. m.	Carpeta de archivos
Eloy2	15/02/2023 01:19 p. m.	Carpeta de archivos
Jorge2do1	16/02/2023 10:29 a. m.	Carpeta de archivos
Jorge2do2	16/02/2023 10:36 a. m.	Carpeta de archivos
Diego2do1	16/02/2023 10:46 a. m.	Carpeta de archivos
Diego2do2	16/02/2023 10:48 a. m.	Carpeta de archivos
Raul2do1	16/02/2023 10:58 a. m.	Carpeta de archivos
Raul2do2	16/02/2023 11:00 a. m.	Carpeta de archivos
Padilla1	16/02/2023 11:47 a. m.	Carpeta de archivos
Padilla2	16/02/2023 11:50 a. m.	Carpeta de archivos
Yissel1	17/02/2023 09:25 a. m.	Carpeta de archivos
Yissel2	17/02/2023 09:28 a. m.	Carpeta de archivos
Daniela1	25/02/2023 10:09 a. m.	Carpeta de archivos
Daniela2	25/02/2023 10:12 a. m.	Carpeta de archivos
Isabela1	25/02/2023 10:24 a. m.	Carpeta de archivos
Isabela2	25/02/2023 10:27 a. m.	Carpeta de archivos
Yolanda1	25/02/2023 10:36 a. m.	Carpeta de archivos
Yolanda2	25/02/2023 10:38 a. m.	Carpeta de archivos

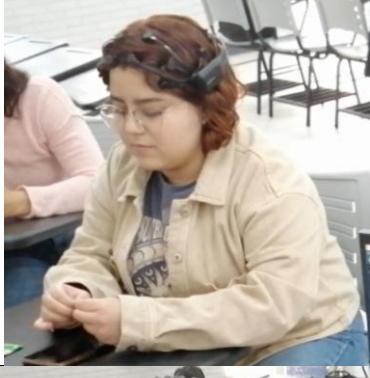
Figura 14.7.1. Evidencia de los conjuntos de datos recopilados en el sistema de archivos.

La experimentación de campo se encuentra documentada de forma visual. A continuación se insertan únicamente capturas de los usuarios durante la experimentación en la tabla 14.7.2 junto con el nombre del perfil que se les asoció para el guardado de carpetas de sus datos.

Tabla 14.7.2. Evidencias de los usuarios voluntarios en la experimentación de campo junto al nombre de perfil recibido.

#	Perfil	Evidencia visual
1	joul	
2	pep	
3	mom	
4	dad	

5	Roman	
6	Hiram	
7	Meli	
8	Oscar	

9	Bere	
10	Andy	
11	Montse	
12	Abraham	

13	Valeria	
14	Juliett	
15	Gus	Evidencia gráfica no disponible
16	Alex	
17	Meza	

18	Fer		
19	Eloy		
20	Jorge2do		
21	Diego2do		

22	Raul2do	
23	Padilla	
24	Yissel	
25	Daniela	Evidencia gráfica no disponible
26	Isabela	Evidencia gráfica no disponible
27	Yolanda	