UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

título del TFG
Documentación Técnica

Presentado por José Luis Pérez Gómez
en Universidad de Burgos — 4 de julio de 2024
Tutor: Bruno Baruque Zanón - Jesús Enrique Sierra García

Índice general

Índice de figuras

Índice de tablas

Apéndice A

Plan de Proyecto Software

A.1. Introducción

La Universidad de Burgos, dentro del área de conocimiento de Ingeniería de Sistemas y Automática, dispone de un interfaz BCI (Brain Computer Interface) para la captación de señales cerebrales. Empleando ese interfaz se han realizado diferentes experimentos que han permitido recoger información de la actividad cerebral mientras los usuarios ejecutaban diferentes tareas cotidianas.

Este Trabajo de Fin de Grado (TFG) tiene como objetivo el análisis de la información obtenida en esos experimentos. Se entrenarán diferentes algoritmos para clasificar la acción realizada por el usuario a partir de las señales generadas por el BCI. Con este propósito, se evaluarán diferentes algoritmos de procesado de señales y de machine/deep learning para la clasificación automática de señales.

Los datos aportados son de tipo EEG (Electroencefalografía) para la realización del TFG son datos referentes a experimentos basados en pulsaciones sobre teclas de un teclado: arriba, abajo, izquierda, derecha.

El análisis de estos datos y su evaluación en diferentes algoritmos esta basada en predecir qué teclas del teclado se han pulsado según las señales captadas con la interfaz BCI.

Para esto no ha habido una planificación como tal registrada en Github, pero sí una progresión definida en los commits de código generados durante la composición del TFG.

A.2. Planificación temporal

En la reunión inicial con los tutores definimos utilizar Python para realizar el código y una serie de aprendizajes básicos para poder acometer el TFG sin problemas.

En las siguientes reuniones se definieron algoritmos y experimentos a realizar con los datos EGG aportados.

Desde el principio del proyecto debido a las circunstancias personales del alumno no se ha podido realizar metodología Scrum, pero sí se produjeron estas lineas temporales:

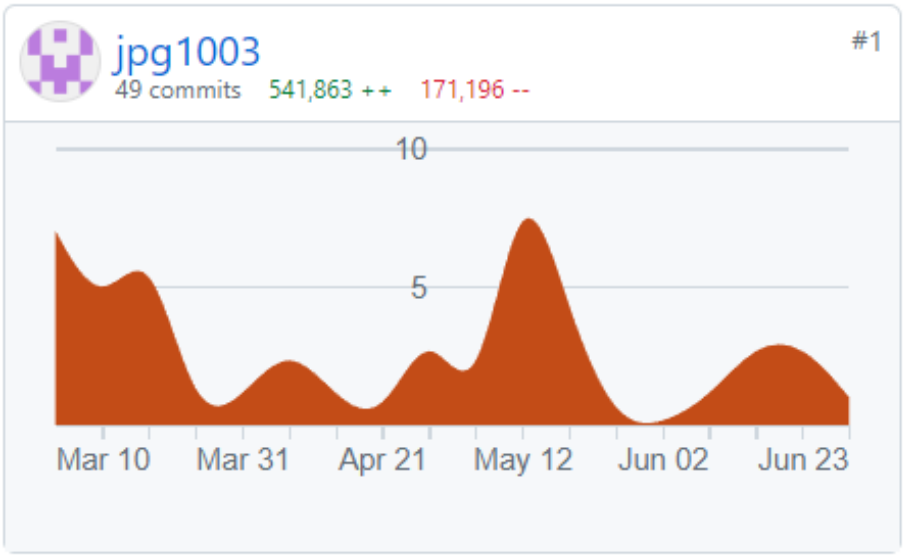

Figura A.1: Commits realizados durante la realización del TFG

Figura A.2: Additions realizados durante la realización del TFG

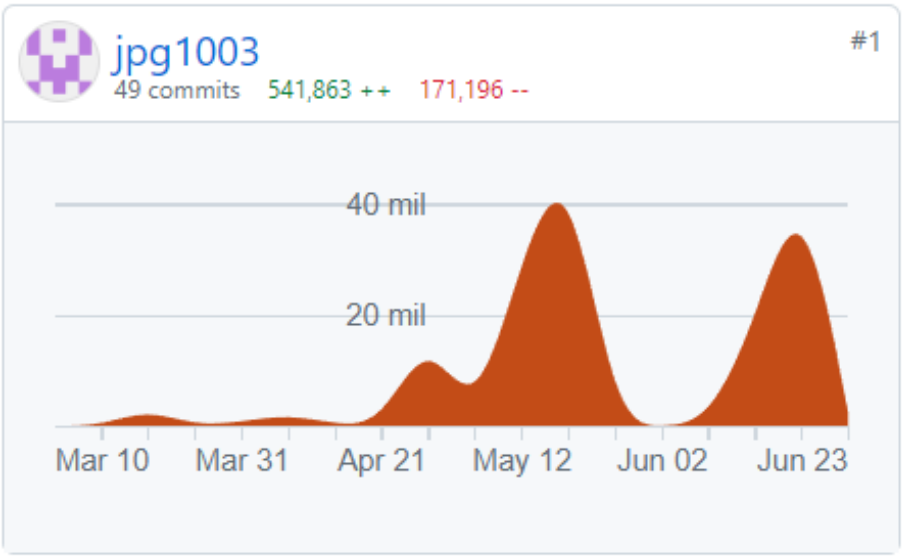

Figura A.3: Deletions realizados durante la realización del TFG

Se pueden dividir en 3 grandes etapas:

- **Etapa de estudio: (Febrero a Marzo)**
- **Etapa de desarrollo: (Abril a Mayo)**
- **Etapa de desarrollo y optimización: (Junio a Julio)**

Etapa de estudio: (Febrero a Marzo)

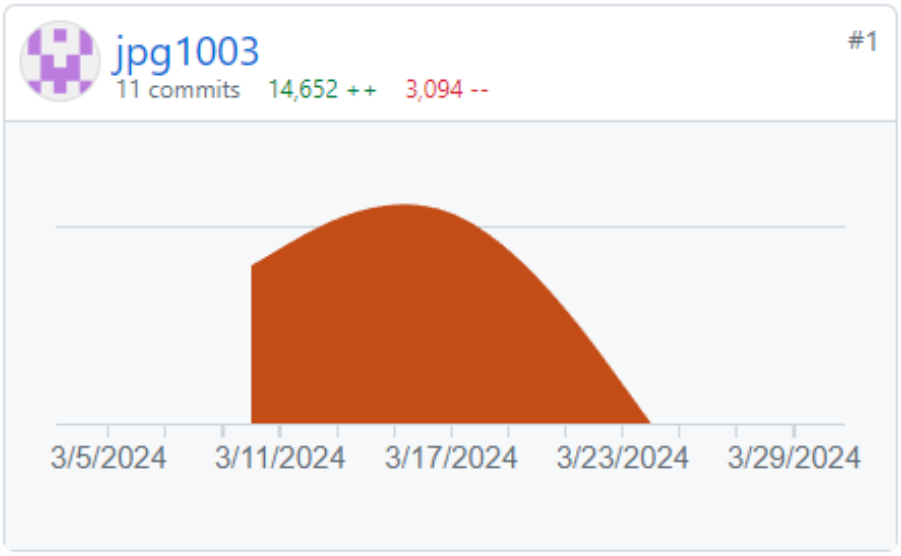

Figura A.4: Commits-Etapa 1

- Realización de cursos online propuestos por los tutores Bruno Baruque y Jesús Enrique Sierra. Tutorial de Inicio de Pandas, Visualización de Datos, Trabajo con series temporales

- Análisis del conjunto de datos, en archivo csv, proporcionado por los tutores.

- Creación esqueleto para la estructura el TFG en Github.

- Definición y creación de los primeros notebooks, principalmente análisis del conjunto de datos.

- Definición y creación de los primeros notebooks de machine learning, al final de la etapa.

- Comentar código e imprimir comentarios en los notebooks.

Los principales problemas o obstáculos que me encontré fueron principalmente los siguientes:

- Tiempo invertido en la realización de los cursos. - Estructuración del código. Me tomo mucho tiempo poder llegar a definir como quería mostrar el código, me decidí por un notebook principal que realizara llamadas al resto de notebooks con códigos mas específicos para cada mo-

delo o experimento a realizar. **- Plotteos y definiciones básicas como normalizar o escalar el conjunto de datos.**

Etapa de desarrollo: (Abril a Mayo)

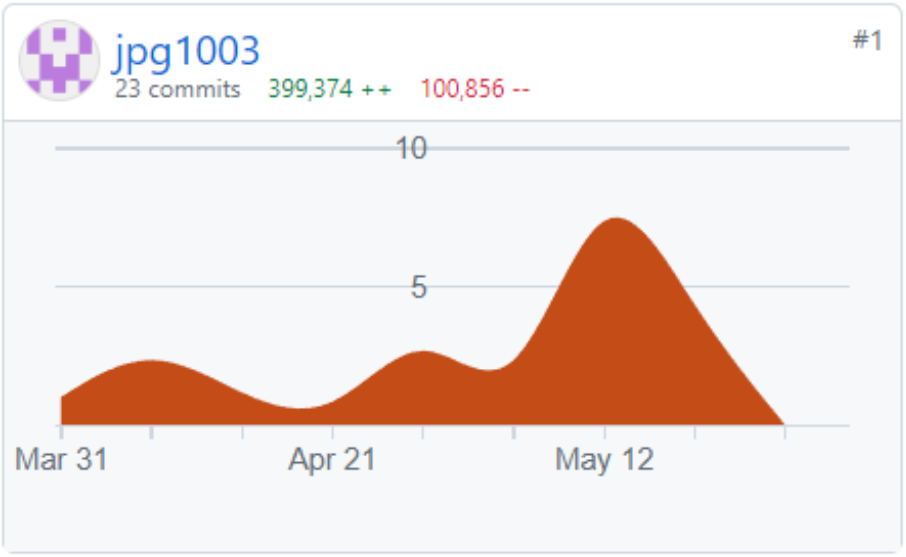

Figura A.5: Commits-Etapa 2

- Cambio de análisis del conjunto de datos, prepocessing.

- Definición y creación notebooks de machine learning.

- Definición y creación notebooks de deep learning.

Los problemas que me encontré en esta etapa fueron:

- Compilación modelos deep learning. Al ejecutar los primeros modelos de deep learning los datos normalizados me proporcionaban errores de compilación con modelos SRNN o LSTM, cambiando a datos escalados y shapeando los modelos pudieron ejecutarse.

- Utilización de ventanas temporales en los modelos deep learning. En esta etapa no supe identificar este requerimiento por parte de los tutores y estuve implementando varias formas de poder utilizar ventanas temporales en el código.

- Utilización de modelos deep learning y callbacks.

- Gráficas y definiciones básicas como normalizar o escalar el conjunto de datos.

Etapa de desarrollo y optimización: (Junio a Julio)

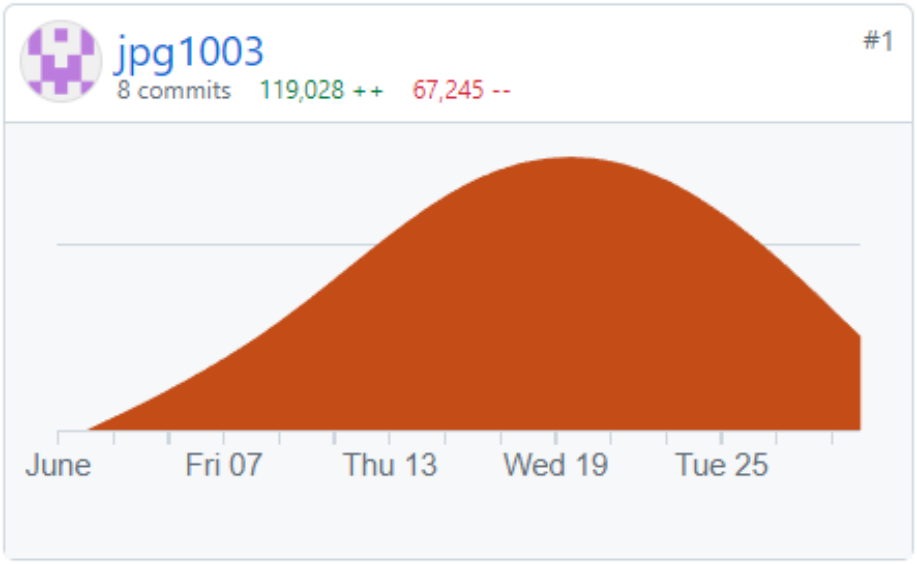

Figura A.6: Commits-Etapa 3

- Añado nuevos gráficos en análisis del conjunto de datos, prepocessing.

- Definición y creación ventanas temporales acordadas con Bruno Baruque.

- Definición y creación nuevos notebooks de deep learning

- Definición y creación nuevos datos sintéticos a través de el aplicativo smote.

- Continuar con el comentado del código e imprimir comentarios en los notebooks.

En la última etapa los problemas que se han acontecido son:

- - Utilización de ventanas temporales en los modelos deep learning. Después de varias algunas reuniones con Bruno Baruque se llego a la defincion correcta para las ventanas temporal en el conjunto de datos.

- Utilización de modelos deep learning y callbacks.

- Utilización datos sintéticos. Definir correctamente esta generacion de datos y poder utilizarlos en los modelos deep learning correctamente.

A.3. Estudio de viabilidad

Viabilidad económica

Viabilidad legal

Apéndice B

Especificación de Requisitos

B.1. Introducción

La Especificación de Requisitos tiene como objetivo definir de manera clara y detallada las necesidades y expectativas del proyecto que consta de un notebook Jupyter definido en Python, el cual invoca a otros notebooks Jupyter secundarios para realizar diversas tareas.

Este documento sirve para asegurar que todas las partes interesadas comprendan y acuerden los objetivos y funcionalidades del sistema a desarrollar.

En un entorno de trabajo basado en notebooks Jupyter, es esencial contar con una especificación precisa que guíe el desarrollo t implementación. Esto facilita la colaboración y la comunicación entre los desarrolladores y usuarios finales.

B.2. Objetivos generales

Los objetivos principales de este TFG son los siguientes:

- **Facilitar el procesamiento de datos EEG.** Generados en la universidad mediante el interfaz BCI.
- **Automatizar tareas repetitivas.** Con la creacion de notebooks secundarios para poder ejecutar las tareas comunes y repetitivas relacionadas con los datos EGG.
- **Mejorar la tasa de acierto en los analisis de modelos** implementando modelos precisos y fiables.

- **Asegurar la accesibilidad y usabilidad en los notebooks.** Los notebooks son herramientas intuitivas y accessibles que aportan esa facilidad a la hora de interacturar.
- **Integracion y extensibilidad.** Al utilizar llamadas a otros notebooks se asegura que se permita la integracion de nuevos notebooks y se asegura posibles nuevas mejoras en el codigo.

B.3. Catálogo de requisitos

Hay dos tipos de requisitos, los funcionales (qué debe hacer el código) y los no funcionales (cómo debe funcionar el código):

Requisitos funcionales

- **RF-001 Cargar archivos con datos EEG:**
  - **Descripción:** El sistema debe permitir cargar archivos de datos EEG para su posterior análisis.
  - **Prioridad:** Alta
  - **Criterios de aceptación:** El sistema debe permitir al usuario poder subir archivos con datos EEG manualmente.
- **RF-002 Preprocesar datos:**
  - **Descripción:** El sistema debe preprocesar los datos EEG cargados de manera manual, estandarizar, escalar, eliminar outliners,... para su posterior análisis. .
  - **Prioridad:** Alta
  - **Criterios de aceptación:** Se ha de poder comprobar que el sistema ha realizado el preprocesado de los datos EEG.
- **RF-003 Visualizar y analizar datos:**
  - **Descripción:** El sistema debe permitir la visualización gráfica de las señales EEG antes y después de ser preprocesadas.
  - **Prioridad:** Alta
  - **Criterios de aceptación:** Se ha de poder comparar los datos EEG antes y después del preprocesado.
- **RF-004 Modelado y validación aprendizaje automático básico:**
  - **Descripción:** El sistema debe permitir entrenar modelos de aprendizaje básicos (KNN, Arboles de decision, random forest)

con diferentes técnicas de validación (Hold-Out,K-Fold, Cross-Validation, Leave-One-Out Cross-Validation) utilizando los datos EEG preprocesados.
- **Prioridad: Alta**
- **Criterios de aceptación:** .

- **RF-005 Modelado y validación aprendizaje automático neuronal:**
  - **Descripción:** El sistema debe permitir entrenar modelos de aprendizaje automático (MLP, SRNN, LSTM) utilizando los datos EEG preprocesados.
  - **Prioridad:** Alta
  - **Criterios de aceptación:** .
- **RF-006 Control de BCI:**
  - **Descripción:** El sistema debe ser capaz de identificar o clasificar con precisión las señales EEG que correspondan a una de las direcciones especificadas (arriba, abajo, izquierda, derecha).
  - **Prioridad: Alta**
  - **Criterios de aceptación:** .
- **RF-007 Almacenamiento y recuperación de resultados:**
  - **Descripción:** El sistema debe permitir almacenar los resultados de los experimentos y sus resultados.
  - **Prioridad: Alta**
  - **Criterios de aceptación:** .
- **RF-008 Integración y Extensibilidad:**
  - **Descripción:** El sistema debe permitir la integración de nuevos modelos de aprendizaje automático sin necesidad de reestructurar significativamente el código existente.
  - **Prioridad: Alta**
  - **Criterios de aceptación:** .

Requisitos no funcionales

- **RNF-001 Rendimiento:**
  - **Descripción:** El sistema donde se ejecuten los notebooks ha de poder manejar la ejecución sin causar demoras significativas.

  - **Criterios de aceptación:** El tiempo de ejecución para la ejecución de cada notebook secundario no ha de ser superior a 15 minutos.

- **RNF-002 Usabilidad:**
  - **Descripción:** Los notebooks han de ser fáciles de entender y usar por los usuarios.
  - **Criterios de aceptación:** La interfaz de los notebooks ha de ser intuitiva y clara.
- **RNF-003 Escalabilidad:**
  - **Descripción:** En la configuración del notebooks principal se debe permitir la adhesión de nuevos notebooks y sus llamadas desde el notebook principal.
  - **Criterios de aceptación:** Poder seguir integrando nuevos notebooks para ser llamados desde el notebook principal sin afectar al funcionamiento del código.

Restricciones

- **R-001:** El sistema debe operar en un entorno Jupyter Notebook.
- **R-002:** Todos los notebooks secundarios deben estar disponibles en el mismo entorno de ejecución que el notebook principal.
- **R-003:** El procesamiento y análisis de datos debe realizarse utilizando bibliotecas compatibles con Python 3.x.

B.4. Especificación de requisitos

Voy a describir cada caso de uso identificado:

CU-001	**Carga archivo con datos EEG**
Actor	Investigador
Versión	1.0
Autor	José Luis Pérez Gómez
Requisitos asociados	RF-001
Descripción	El investigador carga el archivo de datos EEG para su posterior análisis.
Precondición	R-001, R-002, R-003
Acciones	1. El usuario siguiendo los pasos indicados desde el sistema, carga el archivo con datos EEG que quiere analizar.
Postcondición	El usuario podrá visualizar los datos cargados en el sistema
Excepciones	
Importancia	Alta

Tabla B.1: CU-001 Carga archivo con datos EEG.

CU-002	**Preprocesamiento de Datos EEG**
Actor	Investigador
Versión	1.0
Autor	José Luis Pérez Gómez
Requisitos asociados	RF-002
Descripción	El investigador ejecuta el preprocesado de los datos EEG.
Precondición	R-001, R-002, R-003
Acciones	El sistema aplica técnicas de preprocesamiento para limpiar los datos, estandarizar, etc.
Postcondición	La salida de la ejecución no tenga ningún error
Excepciones	
Importancia	Alta

Tabla B.2: CU-002 Preprocesamiento de Datos EEG.

Actor	Investigador
CU-003	**Visualización de datos.**
Versión	1.0
Autor	José Luis Pérez Gómez
Requisitos asociados	RF-003
Descripción	El sistema genera visualizaciones gráficas de las señales EEG antes y después del preprocesado.
Precondición	R-001, R-002, R-003
Acciones	1. El sistema aplica técnicas de impresión por pantalla de los datos a analizar."
Postcondición	Visualizaciones gráficas de los datos EEG.
Excepciones	
Importancia	Alta

Tabla B.3: CU-003 Visualización de datos.

CU-004	**Entrenamiento y validación de modelos de aprendizaje automático básico**
Actor	Investigador
Versión	1.0
Autor	José Luis Pérez Gómez
Requisitos asociados	RF-004
Descripción	El investigador entrena y valida modelos de aprendizaje automático básico utilizando diferentes técnicas de validación con los datos preprocesados EEG.
Precondición	R-001, R-002, R-003
Acciones	1. El sistema entrena y valida los modelos aprendizaje automático básico con los datos preprocesados EEG.
Postcondición	Modelos entrenados y validados con sus respectivas métricas de tasa de acierto y matrices de confusión.
Excepciones	
Importancia	Alta

Tabla B.4: CU-004 Entrenamiento y validación de modelos de aprendizaje automático básico.

CU-005	**Entrenamiento y validación de modelos de aprendizaje automático neuronal**
Actor	Investigador
Versión	1.0
Autor	José Luis Pérez Gómez
Requisitos asociados	RF-005
Descripción	El investigador entrena y valida modelos de deep learning utilizando los datos EEG preprocesados.
Precondición	R-001, R-002, R-003, haber iniciado correctamente el notebook principal (Main) y haber ejecutado todas las celdas anteriores a esta celda
Acciones	1. El sistema entrena y valida los modelos aprendizaje automático neuronal con los datos preprocesados EEG.
Postcondición	Modelos entrenados y validados con sus respectivas métricas de tasa de acierto y matrices de confusión.
Excepciones	
Importancia	Alta

Tabla B.5: CU-005 Entrenamiento y validación de modelos de aprendizaje automático neuronal

Actor	Investigador
CU-006	**Control de BCI**
Versión	1.0
Autor	José Luis Pérez Gómez
Requisitos asociados	RF-006
Descripción	El sistema debe clasificar las señales EEG para determinar la dirección de movimiento (arriba, abajo, izquierda, derecha).
Precondición	R-001, R-002, R-003
Acciones	1. El sistema tras el entrenamiento y validacion de los diferentes modelos debe ser capaz de identificar o clasificar con precisión las señales EEG que correspondan a una de las direcciones especificadas (arriba, abajo, izquierda, derecha)
Postcondición	
Excepciones	
Importancia	Alta

Tabla B.6: CU-006 Control de BCI

Actor	Investigador
CU-007	**Almacenamiento y Recuperación de Resultados de Experimentos**
Versión	1.0
Autor	José Luis Pérez Gómez
Requisitos asociados	RF-007
Descripción	: El sistema almacena y recupera los resultados de los experimentos de análisis de datos EEG.
Precondición	R-001, R-002, R-003
Acciones	1. El investigador solicita la recuperación de resultados de los experimentos realizados y el sistema muestra los resultados de los experimentos.
Postcondición	Resultados de los experimentos almacenados y recuperados para análisis comparativo posterior.
Excepciones	
Importancia	Alta

Tabla B.7: CU-007 Almacenamiento y Recuperación de Resultados de Experimentos

Actor	Desarrollador
CU-008	**Integración y Extensibilidad del sistema**
Versión	1.0
Autor	José Luis Pérez Gómez
Requisitos asociados	RF-008
Descripción	: El desarrollador integra, modifica y extiende todos los modelos de aprendizaje automático en el sistema.
Precondición	R-001, R-002, R-003
Acciones	1. El desarrollador accede al código fuente del sistema, implementa el modelos, actualiza la interfaz de usuario para permitir la selección de los modelos.
Postcondición	Modelos integrados y disponibles para experimentos de análisis de datos EEG.
Excepciones	
Importancia	Alta

Tabla B.8: CU-008 Integración y Extensibilidad del sistema

Apéndice C

Especificación de diseño

C.1. Introducción

La especificación de datos es un componente crítico en el desarrollo de sistemas de información, especialmente cuando se trabaja con conjuntos de datos complejos como los datos EEG.

Mediante BCI, sistema que permite mediante adquisición de señales EEG, se han podido interpretar las señales adquiridas a través de las señales EEG y poder transformarlas en un conjunto de datos para su posterior análisis.

Descripción de los Datos:

El archivo datosEEGTotal.csv contiene los datos facilitados para poder realizar los experimentos para la ejecución del TFG.

Su formato es de tipo CSV con un separador (;) entre los datos que lo componen.

Las características recogidas en el archivo de los datos EEG son las siguientes:

Timestamp, Attention, Meditation, Delta, Theta, LowAlpha, HighAlpha, LowBeta, HighBeta, LowGamma, HighGamma, Signal y Key.

Timestamp: Registro de tiempo para los experimentos, medido en mili-segundos.

Attention: Registra el grado de atención del participante que realiza el experimento.

Meditation: Grado de calma que tendría el individuo.

Delta: Son ondas de baja frecuencia (1 y 4 Hz), están presentes en etapas de sueño profundo, durante una meditación profunda y en pacientes con lesiones cerebrales o con TDAH severo.

Theta: Ondas entre 4 y 8 Hz, se encuentran en estados de calma profunda y sueño R.E.M., están ligadas al aprendizaje, memoria y intuición.

Alpha: Ondas entre 8 y 12 Hz, representan un estado de poca actividad cerebral y se asocian a un estado de calma mental. Divididas en dos señales LowAlpha y HighAplpha

Beta: Se diferencian en LowBeta y HighBeta, su frecuencia esta entre 12 y 35Hz, asociadas a una alta actividad mental.

Gamma: En los datos se diferencia LowGamma y HighGamma, son ondas por encima de 30Hz y suelen aparecer cuando hay una alta concentración o atención

Signal: Podría ser la señal de que aporta la interfaz BCI.

Key: Valores target de lo que el individuo estaba pensando o visualizando durante el experimento.

La transformación de datos o el preprocessing que se ha realizado para poder afrontar los experimentos del TFG han sido los siguientes:

Unificación de características Key: El conjunto de datos tiene varios valores en Key que indican los mismo. LButton y Left.

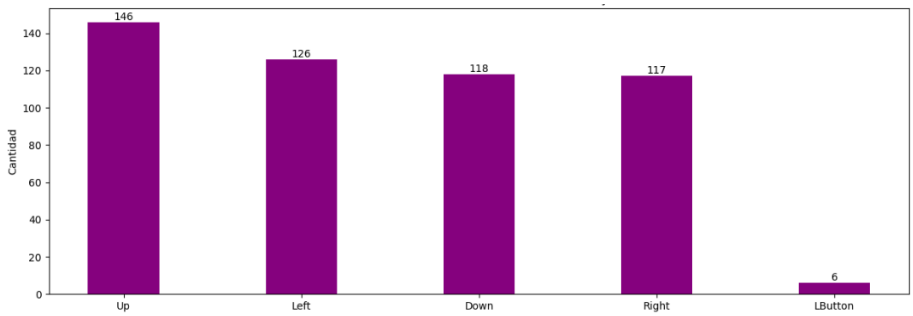

Figura C.1: Cantidad de elementos para la característica Key

División del conjunto de datos: El conjunto de datos tiene cuatro segmentos divididos por su Timestamp, superponiéndose entre ellos. Divido en estos cuatro segmentos para poder realizar experimentos y también dejo

el conjunto de datos sin dividir para realizar experimentos conjuntos a los cuatro segmentos.

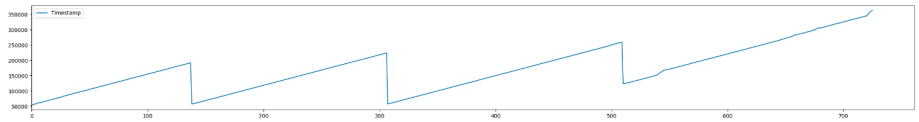

Figura C.2: Análisis característica Timestamp del conjunto de datos

Eliminación de características: Elimino las características Signal por no aportar nada significativo en el conjunto de datos y Timestamp porque no quiero que los datos aportados puedan tener una patrón temporal que haga que los experimentos no sean reales.

Eliminación de outliners: Elimino los outlines (datos atípicos) del conjunto de datos. Utilizo zcore con un umbral de 3. Regla del 68–95–99.7

Escalado de datos: He utilizado la opción de escalar los datos ya que no son datos normales puesto que no tienen una distribución gaussiana en sus datos.

C.2. Diseño de datos

En este trabajo con datos EEG, el diseño de datos consta de:

Diagramas entidad-relación (ERD)

Relaciones y Dependencias:

C.3. Diseño procedimental

En este trabajo he utilizado notebooks Jupyter para analizar el conjunto de datos EEG, por este motivo el diseño arquitectónico resultante ha sido el siguiente:

Arquitectura del Sistema:

La idea principal era que la ejecución de varios modelos de machine y deep learning se pudieran ejecutar de una manera eficiente y que fueran independientes sus ejecuciones. Para esto, el diseño se compone de un notebook jupyer que llama celda a celda a diferentes notebooks en los cuales se desarrolla el código a ejecutar.

Los componentes principales son:

- **Notebook principal**, desde el cual se orquesta todas las ejecuciones de los demás notebooks.
- **Notebooks secundarios preparatorios**, que como su nombre indica, preparan el entorno para que puedan ser ejecutados los experimentos.
- **Notebook subida datos en archivo CSV**, desde esta llamada se sube el archivo con el conjunto de datos a analizar.
- **Notebook secundarios para experimentos** donde se ejecutan los experimentos asociados al TFG.
- **Notebook de resultados** que imprime los resultados de tasa de acierto en los experimentos contra los datos de tipos test.

Las tecnologías y Herramientas utilizadas han sido las siguientes:

Manejo de datos y preprocessing:

- **Pandas:** Para la manipulación y análisis de datos estructurados en DataFrames.
- **NumPy:** Para operaciones numéricas eficientes en matrices, arrays y listas.
- **Scikit-learn y SciPy:** Para escalado de datos, transformación y técnicas de preprocesamiento como StandardScaler y z-score.

Técnicas de validación:

- **Holdout:** División de los datos en conjuntos de entrenamiento, validación y prueba para evaluación inicial.
- **K-Fold Cross-Validation:** Validación cruzada con KFold para evaluar la variabilidad del modelo.
- **Leave-One-Out:** Validación de eno en uno los datos dejando los demás fuera, así se evalúa la robustez del modelo.

Modelos de aprendizaje automático:

- **K-Nearest Neighbors (KNN):** Clasificación basada en la proximidad de los puntos de los datos.
- **Árboles de decisión (Decision Tree):** Utiliza una estructura de árbol para realizar predicciones basadas en decisiones binarias.
- **Random Forest:** Mejora la precisión y reduce el sobreajuste utilizando múltiples árboles de decisión.

Redes Neuronales y Modelos Avanzados:

- **Multi-Layer Perceptron (MLP):** Implementan redes neuronales feedforward. Esta arquitectura de red neuronal no tiene ciclos en sus conexiones neuronales sino que que los datos van en una dirección desde la capa de entrada a la capa de salida. Es el más básico.
- **Simple Recurrent Neural Network (SRNN):** Es un tipo básico de red neuronal que tiene conexiones recurrentes que le permite mantener y utilizar información previa en la secuencia de datos. Es muy parecida a MLP pero en este caso las conexiones entre neuronas si pueden ir hacia detrás.
- **Long Short-Term Memory (LSTM):** Es una red neuronal avanzada en comparación con la recurrente SRNN. La diferencia principal es que puede retener la información relevante durante mas tiempo que SRNN.

Técnicas para la creación de datos sintéticos:

- **SMOTE (Synthetic Minority Over-sampling Technique):** Uso de la biblioteca imblearn para crear ejemplos sintéticos.

Infraestructura utilizada?

C.4. Diseño arquitectónico

Apéndice D

Documentación técnica de programación

D.1. Introducción

Este TFG de análisis de un conjunto de datos EEG se centra en la aplicación de técnicas avanzadas de aprendizaje automático utilizando notebooks Python. El objetivo principal es procesar y analizar datos EEG para identificar patrones y correlaciones significativas que puedan revelar comportamientos específicos en la actividad cerebral a la hora de realizar un experimento el individuo.

El propósito fundamental es aplicar modelos de aprendizaje automático para mejorar la comprensión de las señales EEG, los datos aportados por la universidad de burgos constan de unos experimentos realizados con BCI y varios voluntarios es los que estos usuarios deben visualizar una serie de imagenes compuestas por direcciones, arriba, abajo, derecha e izquierda.

Este trabajo podría tener aplicaciones importantes en tecnologías de interfaz cerebro-computadora (BCI) aplicadas a por ejemplo, control de sillas de ruedas, control de dispositivos electrónicos, realidad virtual, videojuegos, control de robots, etc...

D.2. Estructura de directorios

La estructura de los directorios del trabajo es la siguiente:

- **/documentacion/:** Esta carpeta contiene toda la documentación relacionada con el TFG.
- **/documentacion/imagenes/:** Archivo de imágenes para todo el proyecto, documentación y código.
- **/documentacion/latex:** Documentación en formato latex y PDF.
- **/codigo:** Carpeta para archivar notebooks y datos para la ejecución del código.
- **/codigo/datos/:** Archivo datosEEGTotal.csv con el origen de datos.
- **/codigo/datos/csv/:** Archivos de datos csv auto generados tras las ejecuciones de algunos notebooks.
- **/codigo/notebooks:** Contiene los notebooks Jupyter y por lo tanto, el código.

D.3. Manual del programador

Este manual sera utilizado principalmente por las personas involucradas en futuros cambios, mejoras o nuevos desarrollos en el código.

Para ello, detallo las funciones de cada uno de los componentes:

- **Notebook Principal (1.Main.ipynb):** Crea y ejecuta un entorno virtual. Coordina la ejecución de los notebooks secundarios. Es el punto de entrada para la ejecución del sistema de análisis de datos EEG.
- **Notebooks secundarios:**
  - **2.Bibliotecas.ipynb:** En este notebooks están todas las instalaciones de bibliotecas necesarias para el sistema.
  - **3.Importaciones.ipynb:** Importa las bibliotecas y configuraciones necesarias para el análisis futuro de los datos.
  - **4.1.SubirCSV.ipynb:** Permite la subida de un archivo de datos al Notebook principal.
  - **5.CargaDatos.ipynb:** Realiza un análisis de los datos cargados.
  - **6.Preprocessing.ipynb:** Realiza la limpieza y preparación inicial de los datos EEG..
  - **7.MachineLearning.ipynb:** : Implementa experimentos utilizando metodos como hold-ouy, k-fold cross validation ... para evaluar los modelos de aprendizaje automático básico.
  - **8.1.DeepLearning-MLP.ipynb, 8.2.DeepLearning-SRNN.ipynb, 8.3.DeepLearning-LSTM.ipynb:** Realizan experimentos específicos con modelos de Perceptrón Multicapa, Red Neuronal

Recurrente Simple y Memoria a Corto y Largo Plazo, respectivamente.

- **8.4.DeepLearning-SRNN(SlidingWindows).ipynb, 8.5.DeepLearning-LSTM(SlidingWindows).ipynb:** Realizan experimentos específicos con modelos de Perceptrón Multicapa, Red Neuronal Recurrente Simple y Memoria a Corto y Largo Plazo,respectivamente pero añadiendo ventanas deslizantes.
- **9.1.ProcesadoAumentoDatosSmote.ipynb:** Con smote aumenta y reparte con datos sinteticos la cantidad de datos para el analisis EEG.
- **9.2.DeepLearning-SRNN(AumentoDatos).ipynb, 9.3.DeepLearning-LSTM(AumentoDatos).ipynb:** Implementa experimentos utilizando metodos como hold-ouy, k-fold cross validation ... pero con el aumento de la cantidad de datos con smote.
- **9.4.DeepLearning-SRNN(AumentoDatosSlidingWindows).ipynb, 9.5.DeepLearning-LSTM(AumentoDatosSlidingWindows).ipynb:** Realizan experimentos específicos con modelos de Perceptrón Multicapa, Red Neuronal Recurrente Simple y Memoria a Corto y Largo Plazo,respectivamente pero añadiendo ventanas deslizantes y aumento de la cantidad de datos con smote.
- **10.Resultadosconjuntodatostest.ipynb:** Recopila los resultados de los experimentos y los traslada a varias tablas.

Requisitos de Software:

- **Anaconda**: Para el desarrollo del trabajo he utilizado la solución completa de Anaconda asi he evitado posibles dependencias a la hora de instalar Python o Jupyter Notebook deparados.

Anaconda incluye su propia distribución de Python, por lo que al instalar Anaconda en el sistema, automáticamente se tendrá una instalación de Python junto con una serie de paquetes adicionales útiles para el análisis de datos y la programación científica.

En la ejecución del proyecto también se ejecuta un entorno virtual para evitar dependencias de bibliotecas.

Para desarrollar este proyecto se ha utilizado la plataforma GitHub. La descarga del repositorio se realizara de la siguiente manera:

1. Acceder al repositorio GitHub Acceso repositorio Github

2. Pulsar en Code. (Botón verde)
3. Y a continuación pulsar en Download zip.

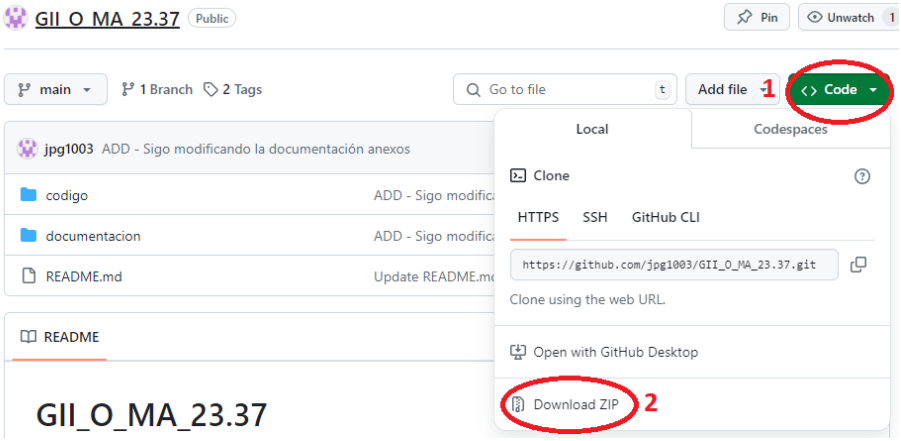

Figura D.1: Descarga repositorio GitHub

4. Llevar el archivo zip descargado a la carpeta que se quiera utilizar para acceder al repositorio.
5. Descomprimir el archivo zip con el nombre de carpeta que se desee.

D.4. Compilación, instalación y ejecución del proyecto

Una vez descargado el repositorio en la carpeta del sistema operativo utilizado se ha de proceder a la instalación del software requerido.

Instalación de software requerido:

- **Anaconda**:
  - Ir al sitio web oficial de Anaconda en Enlace url Anaconda.
  - Descargar la versión más reciente de Python 3.x. para el sistema operativo compatible.
  - Instalar el archivo descargado.

Una vez instalado el paquete de aplicaciones Anaconda, en el sistema operativo aparecerá una aplicación llamada **Jupyter Notebook**:

Ejecutar la aplicación **Jupyter Notebook**, una vez iniciada, se abrirá un navegador de Internet (el que este predeterminado en el sistema operativo)

y mostrara un explorador de archivos por el cual, tendrá que navegar hasta la carpeta descargada del repositorio Github, se debería ver algo parecido a esto:

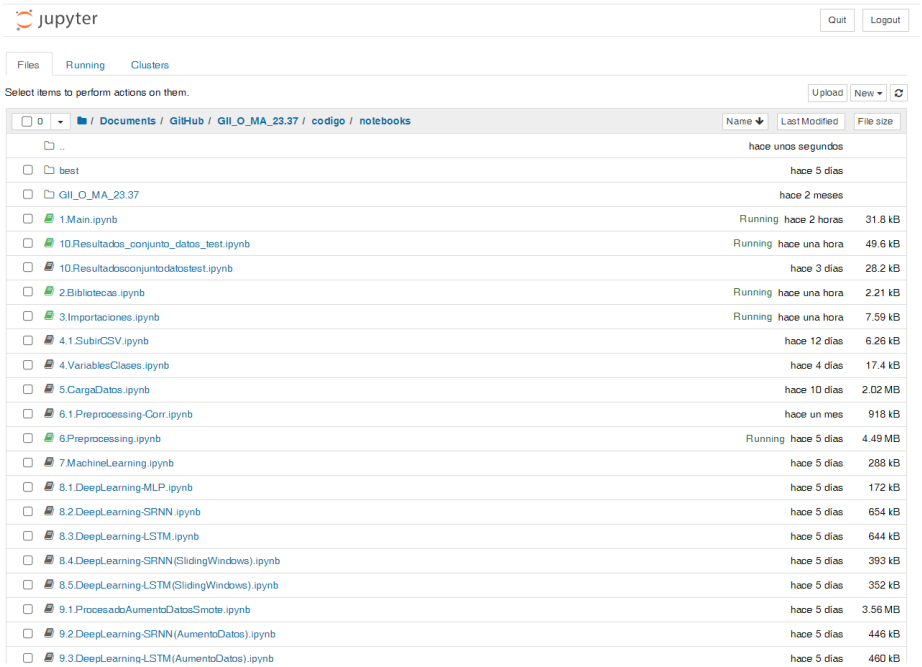

Figura D.2: Listado de Notebooks en Jupyter Notebook

Para ejecutar el proyecto de estudio de datos EEG se puede hacer de dos maneras:

- **Semiautomática**:
  - Abrir el archivo 1.Main.ipynb a través de Jupyter Notebook y ejecutar una a una las celdas de este notebook.

  Manual:
  - Abrir y ejecutar cada uno de los notebooks a través de Jupyter Notebook.
  - Se ha de seguir la numeración marcada por cada uno de los Notebooks, obviando el numero 1.Main.ipynb.
  -

Con la ejecución semiautomática los notebooks se ejecutaran en orden y se quedara toda la información en el mismo notebook. Si fuera de manera

Manual, se tendría más control de lo que se esta ejecutando al estar dividido el código de cada notebook en varias celdas, pero toda la información estaría dividida en cada uno de los notebooks y no en un solo lugar.

D.5. Pruebas del sistema

No se han realizado test con ninguna herramienta porque habria que realizar transformaciones de los notebooks

Apéndice E

Documentación de usuario

E.1. Introducción

E.2. Requisitos de usuarios

E.3. Instalación

E.4. Manual del usuario

Apéndice F

Anexo de sostenibilización curricular

F.1. Introducción

Este anexo incluirá una reflexión personal del alumnado sobre los aspectos de la sostenibilidad que se abordan en el trabajo. Se pueden incluir tantas subsecciones como sean necesarias con la intención de explicar las competencias de sostenibilidad adquiridas durante el alumnado y aplicadas al Trabajo de Fin de Grado.

Más información en el documento de la CRUE https://www.crue.org/wp-content/uploads/2020/02/Directrices_Sosteniblidad_Crue2012.pdf.

Este anexo tendrá una extensión comprendida entre 600 y 800 palabras.