UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

GII-O-MA-23.37 - Análisis de señales EEG

Presentado por José Luis Pérez Gómez
en Universidad de Burgos — 8 de julio de 2024
Tutor: Bruno Baruque Zanón - Jesús Enrique Sierra García

UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

D. Bruno Baruque Zanón, profesor del departamento de nombre departamento, área de nombre área.

Expone:

Que el alumno D. José Luis Pérez Gómez, con DNI 46878665L, ha realizado el Trabajo final de Grado en Ingeniería Informática titulado GII-O-MA-23.37-Análisis de señales EEG.

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 8 de julio de 2024

Vº. Bº. del Tutor:	Vº. Bº. del co-tutor:
D. Bruno Baruque Zanón	D. Jesús Enrique Sierra García

Resumen

El objetivo de este proyecto es desarrollar un sistema de análisis y clasificación de datos EEG (Electroencefalograma) enfocado en la detección y clasificación de intenciones de movimiento (hacia arriba, abajo, derecha e izquierda). Utilizando técnicas avanzadas de aprendizaje automático y procesamiento de datos, este sistema puede servir como base para aplicaciones en interfaces cerebro-computadora (BCI).

Los datos se procesan a través de un flujo de trabajo organizado en un sistema modular, donde se llevan a cabo tareas específicas como la importación de bibliotecas, el preprocesamiento de datos y el entrenamiento de modelos. El proyecto implementa técnicas como el escalado de datos, la validación cruzada y el uso de redes neuronales avanzadas para asegurar la precisión y fiabilidad de los resultados.

Descriptores

Señales EEG, Análisis de datos EEG, preprocesamiento de datos, modelos predictivos, redes neuronales, machine learning, minería de datos, tasa de acierto, matriz de confusión, Python, Jupyter Notebook, visualización de datos ...

Abstract

A **brief** The objective of this project is to develop an EEG (Electroencephalogram) data analysis and classification system focused on the detection and classification of movement intentions (up, down, right and left). Using advanced machine learning and data processing techniques, this system can serve as a basis for applications in brain-computer interfaces (BCI).

Data is processed through a workflow organized in a modular system, where specific tasks such as library import, data preprocessing, and model training are carried out. The project implements techniques such as data scaling, cross-validation and the use of advanced neural networks to ensure the accuracy and reliability of the results.

Keywords

EEG signals, EEG data analysis, data preprocessing, predictive models, neural networks, machine learning, data mining, hit rate, confusion matrix, Python, Jupyter Notebook, data visualization...

Índice general

Índice de figuras

Índice de tablas

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 procesamiento de datos, modelos de machine learning y redes neuronales para la clasificación automática de señales.

Los datos aportados son de tipo EEG (Electroencefalograma) son datos referentes a experimentos basados en acciones de individuas sobre las 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 EEG captadas con la interfaz BCI.

2. Objetivos del proyecto

El objetivo principal de este proyecto es diseñar, implementar y evaluar un sistema integral de análisis y clasificación de señales EEG (Electroencefalograma) centrado en la detección precisa y la clasificación efectiva de acciones de movimiento.

Este sistema se construirá utilizando técnicas avanzadas de machine learning y redes neuronales, con el propósito de desarrollar modelos predictivos robustos capaces de interpretar y categorizar las señales cerebrales asociadas con movimientos específicos, como hacia arriba, abajo, derecha e izquierda.

2.1. Objetivos técnicos

El proyecto se enfocará en varias etapas clave:

- **Adquisición y Preprocesamiento de Datos:**

  Se implementará un flujo de trabajo para la importación, limpieza y estandarización de los datos EEG, asegurando la calidad y consistencia necesarias para el análisis posterior con modelos predictivos.

- **Desarrollo de Modelos Predictivos:**

  Se explorarán y desarrollarán diferentes modelos de machine learning y redes neuronales, adaptados específicamente para el análisis de señales EEG. Esto incluirá el entrenamiento y la optimización de modelos para mejorar la precisión y la generalización. Se emplearán técnicas de ampliación de datos en el conjunto de datos generando nuevos datos sintéticos, otra técnica que se utilizara es la creación de ventanas

deslizantes en el conjunto de datos para garantizar la robustez de los resultados.

- **Validación y Evaluación:**

  Se llevará a cabo una evaluación exhaustiva de los modelos desarrollados, utilizando métricas como la Tasa de Acierto.

En resumen, el proyecto tiene como objetivo técnico avanzar en la comprensión y aplicación de técnicas de machine learning y redes neuronales para el análisis de datos de señales EEG , específicamente en el contexto para la detección precisa de acciones de movimiento.

2.2. Objetivos personales

- **Aprendizaje Avanzado en Machine Learning y Redes Neuronales:**

  Adquirir experiencia práctica en la implementación y optimización de modelos complejos de machine learning y redes neuronales, aplicados específicamente al análisis de señales EEG.

- **Contribución al Avance Científico y Tecnológico:**

  Contribuir al campo de la neurociencia computacional mediante el desarrollo de herramientas y técnicas que puedan mejorar la comprensión y el tratamiento de las señales cerebrales.

- **Desarrollo de Habilidades en Investigación y Análisis:**

  Mejorar habilidades en la investigación científica, la experimentación rigurosa y el análisis crítico de resultados.

- **Aplicación Práctica en Investigación:**

  Facilitar la aplicación de conocimientos teóricos y técnicas avanzadas en un contexto práctico, contribuyendo a la mejora del análisis de señales EEG.

3. Conceptos teóricos

Para llevar a cabo este proyecto, es fundamental comprender y aplicar diversos conceptos teóricos que sustentan tanto la adquisición y procesamiento de datos EEG como el desarrollo de modelos predictivos eficientes y precisos.

A continuación se describen varios conceptos teóricos:

3.1. Electroencefalograma

El Electroencefalograma (EEG) es una técnica no invasiva utilizada para registrar la actividad eléctrica del cerebro.

Las señales EEG son capturadas mediante electrodos colocados en la cabeza, que detectan los cambios en el potencial eléctrico generados por la actividad neuronal.

Estas señales reflejan la actividad de gran cantidad de neuronas y son fundamentales para el estudio de diversas funciones cerebrales.

Figura 3.1: Ejemplo colocación electrodos para una interfaz BCI. [7]

Las principales señales cerebrales en un EEG se componen de diferentes ondas que se clasifican según su frecuencia.

Estas ondas reflejan distintos estados de actividad cerebral y son las que se utilizan para el análisis de datos de señales EEG.

- **Ondas Delta:**
  - **Frecuencia:** 0.5 - 4 Hz.
  - **Asociación:** Sueño profundo y estados de inconsciencia.
- **Ondas Theta:**
  - **Frecuencia:** 4 - 8 Hz
  - **Asociación:** Estados de somnolencia, meditación, y el sueño ligero.
- **Ondas Alfa:**.
  - **Frecuencia:** 8 - 13 Hz
  - **Asociación:** Estado de relajación y vigilia tranquila con los ojos cerrados.
- **Ondas Beta:**

  - **Frecuencia:** 13 - 30 Hz
  - **Asociación:** Estados de alerta, concentración activa, y actividad mental intensa.

- **Ondas Gamma:**
  - **Frecuencia:** 30 - 100 Hz
  - **Asociación:** Procesos cognitivos complejos, como la percepción consciente y el procesamiento de la información sensorial.

3.2. Conjunto de datos

Un conjunto de datos es una colección de datos organizados en un formato estructurado, generalmente en forma de una tabla donde cada fila representa una observación o instancia, y cada columna representa una característica o variable.

Un conjunto de datos puede dividirse en dos secciones principales: las características (features) y los targets (objetivos).

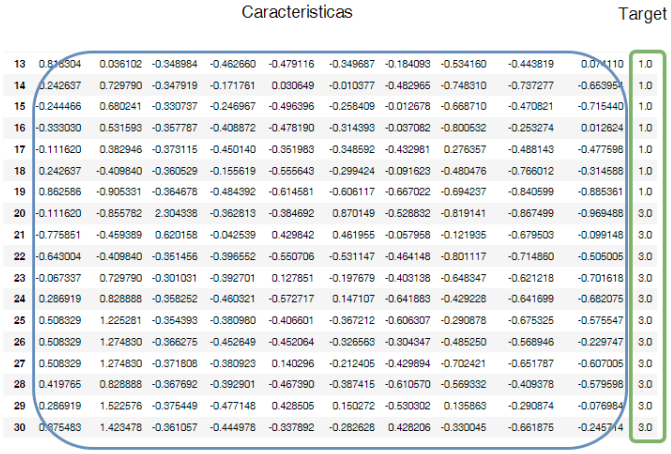
- **Característica (features):**

  Las características son las variables que se utilizan para predecir o clasificar el target. Estas variables pueden ser de diferentes tipos: numéricas, categóricas, de texto, etc...

- **Target (objetivo):**

  El target es la variable que se desea predecir o clasificar utilizando las características.

  En tareas de clasificación, el target es una etiqueta o categoría que representa la clase a la que pertenece cada observación o instancia.



13	0.818304	0.036102	-0.348984	-0.462660	-0.479116	-0.349687	-0.184093	-0.534160	-0.443819	0.074110	1.0
14	0.242637	0.729790	-0.347919	-0.171761	0.030649	-0.010377	-0.482965	-0.748310	-0.737277	-0.653961	1.0
15	-0.244466	0.680241	-0.330737	-0.246967	-0.496396	-0.258409	-0.012678	-0.668710	-0.470821	-0.715440	1.0
16	-0.333030	0.531593	-0.357787	-0.408872	-0.478190	-0.314393	-0.037082	-0.800532	-0.253274	0.012624	1.0
17	-0.111620	0.382946	-0.373115	-0.450140	-0.351983	-0.348592	-0.432981	0.276357	-0.488143	-0.477598	1.0
18	0.242637	-0.409840	-0.360529	-0.155619	-0.555643	-0.299424	-0.091623	-0.480476	-0.766012	-0.314588	1.0
19	0.862586	-0.905331	-0.364678	-0.484392	-0.614581	-0.606117	-0.667022	-0.694237	-0.840599	-0.885361	1.0
20	-0.111620	-0.855782	2.304338	-0.362813	-0.384692	0.870149	-0.528832	-0.819141	-0.867499	-0.969488	3.0
21	-0.775851	-0.459389	0.620158	-0.042539	0.429842	0.461955	-0.057958	-0.121935	-0.679503	-0.099148	3.0
22	-0.643004	-0.409840	-0.351456	-0.396552	-0.550706	-0.531147	-0.464148	-0.801117	-0.714860	-0.505005	3.0
23	-0.067337	0.729790	-0.301031	-0.392701	0.127851	-0.197679	-0.403138	-0.648347	-0.621218	-0.701618	3.0
24	0.286919	0.828888	-0.358252	-0.460321	-0.572717	0.147107	-0.641883	-0.429228	-0.641699	-0.682075	3.0
25	0.508329	1.225281	-0.354393	-0.380980	-0.406601	-0.367212	-0.606307	-0.290878	-0.675325	-0.575547	3.0
26	0.508329	1.274830	-0.366275	-0.452649	-0.452064	-0.326563	-0.304347	-0.485250	-0.568946	-0.229747	3.0
27	0.508329	1.274830	-0.371808	-0.380923	0.140296	-0.212405	-0.429894	-0.702421	-0.651787	-0.607005	3.0
28	0.419765	0.828888	-0.367692	-0.392901	-0.467390	-0.387415	-0.610570	-0.569332	-0.409378	-0.579598	3.0
29	0.286919	1.522576	-0.375449	-0.477148	0.428505	0.150272	-0.530302	0.135863	-0.290874	-0.076984	3.0
30	0.975483	1.423478	-0.361057	-0.444978	-0.337892	-0.282628	0.428206	-0.330045	-0.661875	-0.245714	3.0

Figura 3.2: Ejemplo conjunto de datos [2]

3.3. Preprocesado de datos

El preprocesamiento de datos EEG es una fase crucial en el análisis de señales EEG, ya que asegura la calidad y la fiabilidad de los datos antes de su análisis y clasificación. A continuación se describen las técnicas específicas que se han aplicado en este proyecto para el preprocesamiento de los datos EEG:

- **Modificación datos targets por perdida o unificación:**

  Se ha realizado una unificación de los datos targets, para esto se han identificado que valores únicos tenia esta característica del conjunto de datos y una vez identificado unificar los que indican los mismo pero con otro valor. Como ejemplo: Left y LButton, son diferentes pero indican mismo valor.

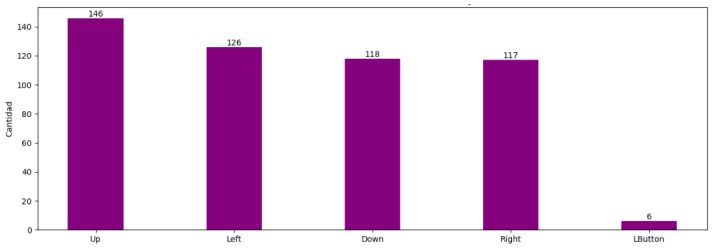


Figura 3.3: Ejemplo valores de los datos target

Los datos perdidos en target son aquellos que no indican ningún movimiento. Aunque no indiquen movimiento alguno se les ha de aplicar un valor para que el análisis de los datos sea correcto.

```
RangeIndex: 726 entries, 0 to 725
Data columns (total 13 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Timestamp   726 non-null    int64
 1   Attention   726 non-null    int64
 2   Meditation  726 non-null    int64
 3   Delta       726 non-null    int64
 4   Theta       726 non-null    int64
 5   LowAlpha    726 non-null    int64
 6   HighAlpha   726 non-null    int64
 7   LowBeta     726 non-null    int64
 8   HighBeta    726 non-null    int64
 9   LowGamma    726 non-null    int64
 10  HighGamma   726 non-null    int64
 11  Signal      726 non-null    int64
 12  Key         513 non-null    object
dtypes: int64(12), object(1)
memory usage: 73.9+ KB
```

Figura 3.4: Ejemplo valores perdidos en un conjunto de datos

- **Eliminación de señales del conjunto de datos:**

  Se eliminan señales que no son necesarias para el análisis porque no aportan información relevante sobre la actividad cerebral.

- **Eliminación de outliners:**

  Los outliers son valores atípicos que pueden distorsionar los resultados del análisis de datos. En el contexto de EEG, los outliers pueden surgir debido a artefactos o errores en la adquisición de datos.

  - **Detección de Outliers:**

    Se aplican técnicas estadísticas para identificar valores que se desvían significativamente del resto de los datos.

El método utilizado ha sido z-core que mide la distancia de un valor desde la media del conjunto de datos en términos de desviación estándar.

$$zcore = (dato - media.datos)/desviacion.estándar$$

Para la evaluación e identificación de posibles valores que sean outliners se ha utilizado la regla empírica o regla 68-95-99.7. [8] Usando esta regla, se consideran outliers los datos que tienen un z-score mayor a 3 o menor a -3, ya que caen fuera del rango en el que se encuentra el 99.7 por ciento de los datos.

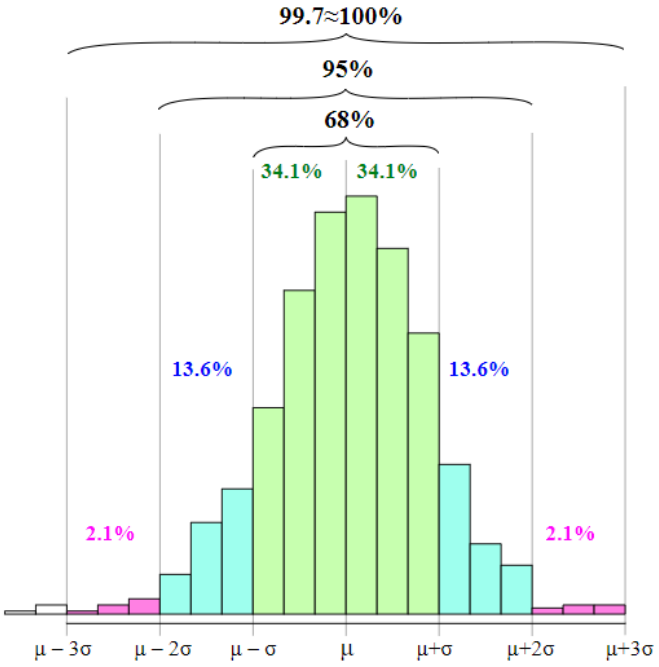


Figura 3.5: Regla 68–95–99.7 [8]

- **Escalado del conjunto de datos:**

  Con el escalado de datos se prepararan los datos EEG antes de aplicar algoritmos o modelos de aprendizaje automático. El objetivo es normalizar las amplitudes de las señales para que se encuentren dentro de un rango común, facilitando la comparación y el análisis.

  Se ha aplicado un escalado en rangos de -1 a 1 para que la escala sea uniforme.

Este escalado se ha realizado con **StandardScaler()** que es es una herramienta de preprocesamiento de datos proporcionada por la biblioteca **scikit-learn** en Python.

Se utiliza para estandarizar las características de un conjunto de datos de manera que tengan una media igual a 0 y una desviación estándar igual a 1.

Este proceso es muy beneficioso ya que asegura que todas las características contribuyan de manera equitativa al modelo y mejora la convergencia de algunos algoritmos de optimización.

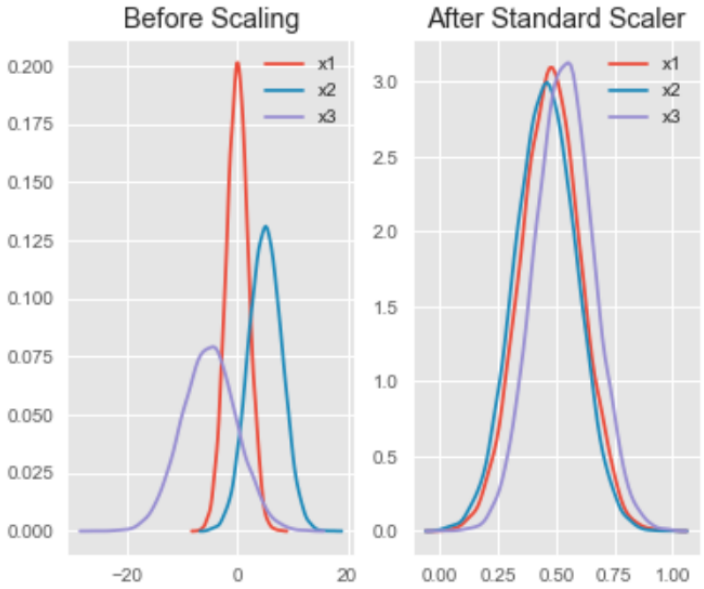


Figura 3.6: Ejemplo utilización StandardScaler() [5]

- **Segmentación de Datos:** Se ha realizado segmentación de datos para dividir las señales EGG del conjunto de datos en conjunto de datos mas pequeños, esto se basa en la característica TimeStamp, con la cual se pudo identificar que la linea temporal se repetía o solapaba en algunas instancias.

  De esta manera se pueden realizar experimentos de análisis sobre datos de diferentes individuos independientemente unos de otros.

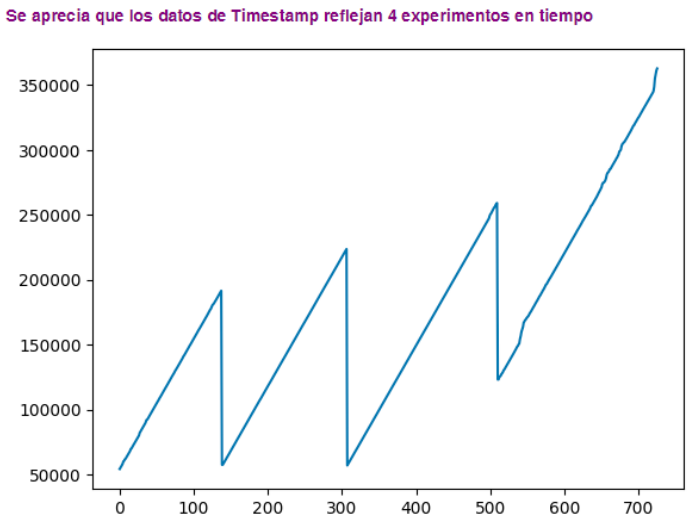


Figura 3.7: Datos Timestamp conjunto de datos

3.4. Partición de datos para el entrenamiento

En machine learning y redes neuronales, es fundamental dividir el conjunto de datos en tres subconjuntos: entrenamiento (train), validación (val) y prueba (test).

Cada uno de estos subconjuntos tiene un propósito específico y ayuda a garantizar que el modelo generalice bien a datos nuevos y no procesados con anterioridad en la ejecución del entrenamiento de los modelos.

Para lograr la división del conjunto de datos en subconjuntos he utilizado la función **train-test-split** de scikit-learn. Utilizándola en dos pasos.

1. División del conjunto de datos en Test y Train-intermedio. 90 por ciento para Train-intermedio. 2. División de Train-intermedio en Val y Train. 90 por ciento para Train.

Quedando la división aproximada en tanto por ciento, de la siguiente manera: 80 para Train, 10 para Val, 10 para Test

Figura 3.8: Ejemplo particiones del conjunto de datos [2]

- **Subconjunto de Entrenamiento (Train):**

  El subconjunto de entrenamiento se utiliza para ajustar los parámetros del modelo. Durante el entrenamiento, el modelo aprende patrones y relaciones en los datos de este subconjunto.

  En los experimentos para los diferentes modelos de aprendizaje será del 80 por ciento de los datos del conjunto de datos inicial.

- **Subconjunto de Validación (Val):**

  El subconjunto de validación se utiliza para ajustar los hiperparámetros del modelo y para evaluar su rendimiento durante el proceso de entrenamiento.

  Ayuda a prevenir el sobreajuste (overfitting).

- **Subconjunto de Prueba (Test):**

  El subconjunto de prueba se utiliza para evaluar el rendimiento final del modelo después de haber sido entrenado y ajustado utilizando los conjuntos de entrenamiento y validación.

  Se utilizan estos datos al final de la ejecucion del modelo.

  Este conjunto proporciona una medida final de la capacidad del modelo para generalizar a datos no vistos o procesados con anterioridad en los entrenamientos de los modelos, lo cual es crucial para determinar si el modelo es apto para su implementación en un entorno real.

3.5. Sobreajuste (overfitting)

El sobreajuste (overfitting) es un problema común en el aprendizaje automático y las redes neuronales, donde un modelo aprende demasiado bien los detalles y el ruido del conjunto de datos de entrenamiento, hasta el punto de que su rendimiento en datos nuevos y no vistos con anterioridad se deteriora.

El modelo se ajusta tan estrechamente a los datos de entrenamiento que pierde la capacidad de generalizar a otros conjuntos de datos.

- **Detección del sobreajuste:**

  Una indicación clara de sobreajuste es cuando la tasa de acierto del modelo en el conjunto de entrenamiento sigue mejorando mientras que la tasa de acierto en el conjunto de validación se estanca o empeora.

  Prevenir el sobreajuste:

  - **Dropout:** En redes neuronales, se desactivan algunas neuronas durante el entrenamiento para evitar que el modelo dependa demasiado de características específicas.

    Se utiliza después de las capas de entrada y capas ocultas de las redes neuronales y los valores recomendados oscilan entre el 20 y 50 por ciento de neuronas desconectadas. De esta forma el aprendizaje no pueda tender al sobreajuste.

    Para este proyecto utilizare como valor el 30 por ciento.
  - **Aumento de datos:** Generar nuevas muestras de datos sintéticos a partir de las existentes mediante técnicas como la creación de ventanas deslizantes en series temporales o creación de nuevos datos sintéticos.
  - **Validaciones cruzadas:**

    Dividir los datos en subconjuntos y entrenar el modelo varias veces, utilizando subconjuntos para entrenamiento y validación en cada iteración.
  - **Uso de callbacks en redes neuronales:**

    Los callbacks permiten realizar acciones y monitorizar el entrenamiento en tiempo real, proporcionando una manera eficiente de controlar el modelo en su entrenamiento.

    Hay varios tipos de callback pero para el proyecto se utilizaran los siguientes:

    - **EarlyStopping:**
      Este callback detiene el entrenamiento cuando una métrica monitoreada deja de mejorar.
      En los modelos que se han implementado se utilizara la métrica **val-loss** (Pérdida (loss)) que es una medida de lo bien o mal que el modelo está haciendo sus predicciones en relación con los valores reales.

Es una función matemática que calcula la discrepancia entre las predicciones del modelo y las etiquetas verdaderas.

- **ReduceLROnPlateau:**
  Monitoriza la misma métrica **val-loss** y reduce la tasa de aprendizaje en un factor de 0.1 si no mejora después de 5 épocas o epochs.
  Época (epoch) es una pasada completa por todo el conjunto de datos de entrenamiento. Durante una epoch, el modelo ve y procesa cada instancia del conjunto de datos una vez. En un modelo de red neuronal se definen el numero de epochs que va a tener el entrenamiento del modelo. Para este proyecto seria de 30 épocas.
- **ModelCheckpoint:**
  Guarda el mejor rendimiento que ha tenido durante las épocas el entrenamiento, este callback esta basado en la métrica **val-loss** y se guarda en un archivo .keras. para después poder ser utilizado al termino del entrenamiento.

3.6. Machine Learning

El uso de técnicas de machine learning en el análisis de datos EEG permite la detección y clasificación precisa de patrones en las señales cerebrales. En este proyecto, se han implementado diversos métodos de validación y algoritmos de clasificación para garantizar la robustez y fiabilidad de los modelos predictivos.

A continuación, se describen las técnicas y algoritmos utilizadas:

- **Técnicas de Validación:**

  - **Holdout:**
    El método holdout implica dividir el conjunto de datos tal y como se ha descrito en la sección: "Partición de datos para el entrenamiento".

    Esta técnica es simple y rápida, los datos han de estar distribuidos uniformemente entre las particiones.

    Para que los datos estén distribuidos uniformemente dentro de los tres subconjuntos he utilizado el parámetro **stratify**, a este proceso se conoce como .estratificaciónz asegura que el balanceo de los datos del conjunto original se mantenga en los subconjuntos de

Train, Val y Test. Es importante porque permite que los modelos se entrenen y evalúen de manera más representativa.

- **K-Fold Cross-Validation:**

  La validación cruzada k-fold divide el conjunto de datos en k subconjuntos (folds) de tamaño aproximadamente igual.

  El modelo se entrena k veces, utilizando k-1 folds para entrenamiento y uno para prueba en cada iteración.

  Este proceso se repite k veces, asegurando que cada fold se utilice como conjunto de prueba una vez.

- **Leave-One-Out Cross-Validation:**

  Es una técnica de validación cruzada donde k es igual al número de muestras en el conjunto de datos.

  En cada iteración, una sola muestra se utiliza como conjunto de prueba, y el resto se utiliza para entrenamiento.

  Este método garantiza que cada muestra se evalúe como conjunto de prueba, proporcionando una evaluación precisa del modelo.

  Esta técnica puede ser computacionalmente costosa para conjuntos de datos grandes.

- **Algoritmos de Clasificación:**

  - **K-Nearest Neighbors (KNN):**

    KNN es un algoritmo de clasificación basado en instancias que asigna una clase a una muestra en función de la mayoría de sus k vecinos más cercanos.

    La distancia entre muestras se calcula utilizando la distancia euclidiana.

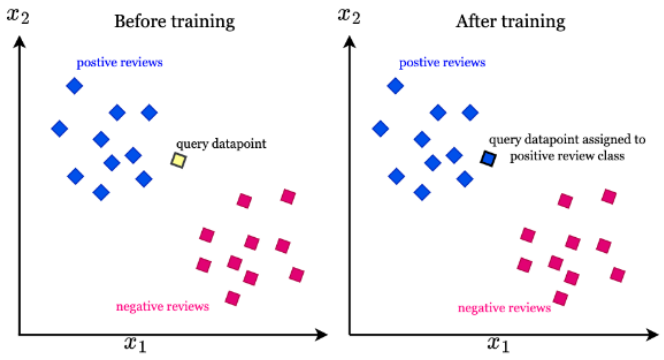


Figura 3.9: Uso KNN. [4]

- **Árboles de Decisión:**
  Los árboles de decisión son modelos de clasificación que dividen los datos en subconjuntos basados en valores de características, organizados en una estructura de árbol.
  Cada nodo del árbol representa una característica, cada rama una decisión, y cada hoja una clase.

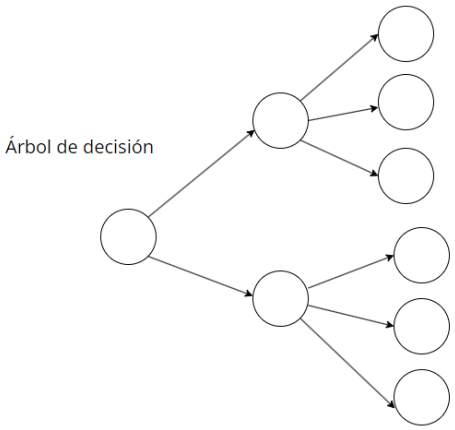


Figura 3.10: Uso arboles de decisión. [3]

- **Random Forest:**
  Random Forest es un conjunto de árboles de decisión que utiliza el bagging (bootstrap aggregating) para mejorar la precisión y reducir el sobreajuste.

Cada árbol del bosque se entrena con una muestra aleatoria del conjunto de datos, y las predicciones se combinan para obtener el resultado final.

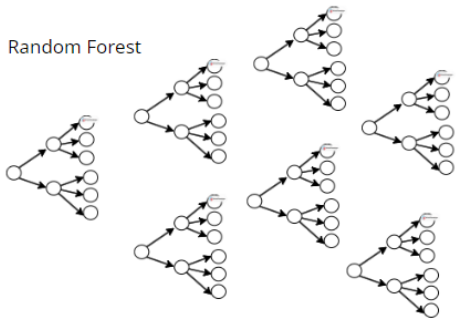


Figura 3.11: Uso random forest. [3]

3.7. Redes neuronales

Las redes neuronales son modelos de aprendizaje profundo que simulan el funcionamiento del cerebro humano para reconocer patrones complejos en los datos.

En este proyecto, se han utilizado varios tipos de redes neuronales para el análisis de datos EEG:

- **Multilayer Perceptron (MLP):**

  Un MLP es una red neuronal artificial feedforward con una o más capas ocultas entre la capa de entrada y la capa de salida.

  Consta de una capa de entrada, una o más capas ocultas y una capa de salida. Cada neurona en una capa está conectada a todas las neuronas de la siguiente capa. Las capas ocultas permiten al MLP aprender representaciones no lineales de los datos. Tienen solo una dirección de conexión entre capas, hacia delante.

  El MLP se entrena utilizando un algoritmo que ajusta los pesos de las conexiones neuronales para minimizar el error de predicción.

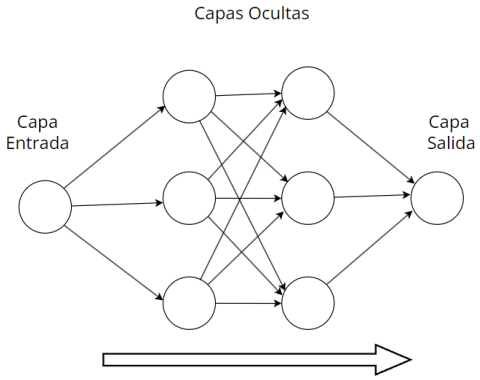


Figura 3.12: Red neuronal MLP. [3]

- **Recurrent Neural Network (RNN):**

  Una SRNN es un tipo de red neuronal en la que las conexiones entre las neuronas forman un ciclo, lo que permite que la información se mantenga a lo largo del tiempo. Las neuronas de diferentes capas están conectadas hacia adelante o hacia atrás.

  Esta red tiene una capa de entrada, una o mas capas recurrentes ocultas y una capa de salida.

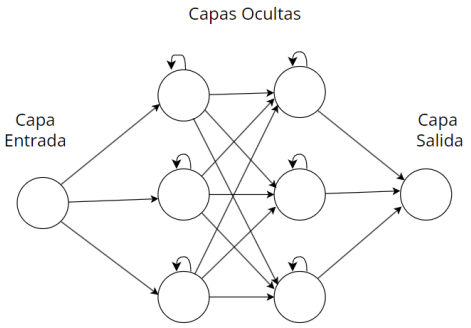


Figura 3.13: Red neuronal RNN. [3]

- **Long Short-Term Memory (LSTM):**

  Las LSTM son un tipo de red neuronal recurrente diseñada para aprender dependencias a largo plazo en datos secuenciales. Utilizan celdas de memoria que pueden mantener información durante largos períodos.

La arquitectura de esta red consiste en una capa de entrada, una o más capas LSTM y una capa de salida.

La red neuronal es muy parecida a la de las RNN pero las LSTM añaden:

Celdas de memoria que sirven para escribir, leer o guardar datos, estas celdas están controladas por las puertas.

Puertas que se dividen en puertas de entrada, de olvido y de salida que sirven para el control del flujo de datos, estas puertas regulan el estado de las celdas de memoria.

3.8. Ventana Deslizante

La ventana deslizante es una técnica que consiste en tomar un subconjunto de datos de un conjunto mayor, moviéndose a través del conjunto de datos por pasos fijos. Este subconjunto se llama "ventana". Cada vez que la ventana se desliza, se incluye un nueva ventana de datos que analizar. Para este proyecto se han creado ventanas deslizantes unificando temporalmente los datos por cada target, Key u objetivo.

Esto ayuda a manejar y analizar datos secuenciales, permitiendo al modelo enfocarse en patrones locales en los datos mientras mantiene la referencia a un target, Key u objetivo constante. Esto generara varios conjunto de ventanas, uno por cada objetivo.

Figura 3.14: Ejemplo ventanas temporales sobre targets. [2]

3.9. Generación de datos sintéticos

La generación de datos sintéticos en un conjunto de datos se refiere al proceso de crear nuevos datos que imitan las características estadísticas y estructurales del conjunto de datos original.

Este enfoque es útil en diversas situaciones, como cuando se quiere aumentar el tamaño del conjunto de datos para entrenar modelos de aprendizaje automático o equilibrar clases desproporcionadas.

El método utilizado es sobremuestreo (Oversampling), principalmente replica y ajusta datos existentes para crear nuevas muestras. No elimina las muestras los datos originales sino que crea nuevos datos.

3.10. Matriz de confusión

Una matriz de confusión es una herramienta en la evaluación de modelos de clasificación en aprendizaje automático. Proporciona una visualización de la calidad del rendimiento de un modelo al comparar las predicciones del modelo con los valores reales de los datos de prueba.

- **Verdaderos Positivos (VP):**

  Son los casos donde el modelo predijo correctamente la clase positiva.

- **Falsos Negativos (FN):**

  Son los casos donde el modelo predijo incorrectamente la clase negativa cuando en realidad era positiva.

- **Falsos Negativos (FP):**

  Son los casos donde el modelo predijo incorrectamente la clase positiva cuando en realidad era negativa.

- **Verdaderos Negativos (VN):**

  Representan los casos donde el modelo predijo correctamente la clase negativa.

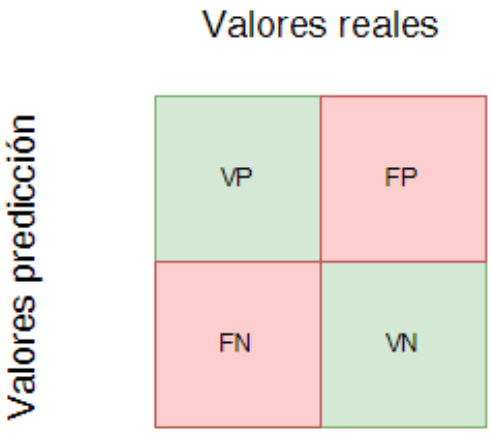


Figura 3.15: Matriz de confusión. [2]

3.11. Tasa de acierto

La tasa de acierto es una métrica que proporciona una visión general del rendimiento del modelo. Sin embargo, su interpretación debe realizarse con precaución, especialmente en casos donde las clases están desbalanceadas. Para este proyecto las clases están balanceadas.

$$Tasadeacierto = (VP + VN)/VP + FN + FP + VN$$

Tasa de acierto es el número de predicciones correctas entre número total de predicciones.

4. Técnicas y herramientas

En esta sección de la memoria, además de detallar las técnicas metodológicas y las herramientas de desarrollo, se incluyen explicaciones sobre los diferentes experimentos realizados y las herramientas específicas utilizadas, como Jupyter Notebooks y Python.

A continuación, se profundiza en estos aspectos:

Técnicas metodológicas

Conjuntos de datos a analizar

Se han creado cuatro conjuntos de datos, cada uno conteniendo señales EEG capturadas de diferentes sesiones o condiciones experimentales.

Estos conjuntos se han separado basándose en la característica TimeStamp dentro del conjunto de datos global.

Estos conjuntos están etiquetados por Segmentos, numerados del 1 al 4 y organizados de acuerdo con las especificaciones individuales para cada participante en el estudio.

Además de los conjuntos de datos individuales por sesión o condiciones experimentales, se han creado dos conjuntos de datos que combinan las señales EEG de todos las sesiones capturadas.

El primer conjunto se crea después de haber sido escalados los segmentos individualmente y se llama All Segmentos after.

El segundo conjunto se crea escalando todo el conjunto de datos sin haber separado los segmentos individualmente con anterioridad. All Segmentos before.

Un primer enfoque que permite analizar las diferencias entre los modelos y como clasifican los datos por sesiones o condiciones experimentales individualmente.

Y un segundo enfoque, mas atractivo, en un contexto unificado, proporcionando una visión generalizada de las clasificaciones de los modelos para varias sesiones o condiciones experimentales que aporta una visión mas real de estos análisis.

Total 6 conjuntos de datos para experimentar.

4 para Segmentos individuales y 2 para conjuntos de datos totales.

Experimento comparativo

En este experimento se compararon los modelos explicados en la sección Conceptos teóricos, machine learning y redes neuronales.

Se llevaron a cabo ejecuciones utilizando holdout, k-fold cross-validation , leave-one-out cross-validation, MLP, RNN y LSTM para evaluar la robustez y la generalización de los modelos desarrollados.

Se realizaron los experimentos con los 6 conjuntos de datos antes mencionados y se ha avaluado la tasa de acierto de los diferentes modelos.

Experimento con ventanas deslizantes

Este experimento solo se puede realizar con los dos conjuntos de datos totales puesto que en las particiones de datos asociadas a ventanas temporales influye la temporalidad y si se utilizan conjunto de datos de Segmentos individuales los datos no estarían balanceados y no estarían representados todos los datos en cada una de las particiones de datos.

En este experimento se comparan los dos modelos mas complejos de redes neuronales RNN y LSTM y se avaluó la generalización para estos datos asociados a diferentes sesiones de captura de datos EEG.

La evaluación de los modelos ha sido mediante tasa de acierto.

Experimento con creación de datos sintéticos

Este experimento solo se puede realizar con los dos conjuntos de datos totales puesto que en las particiones de datos asociadas a ventanas temporales influye la temporalidad y si se utilizan conjunto de datos de Segmentos individuales los datos no estarían balanceados y no estarían representados todos los datos en cada una de las particiones de datos.

al igual que en el anterior experimento se comparan los modelos dos modelos mas complejos de redes neuronales RNN y LSTM y se avaluó la generalización para estos datos asociados a diferentes sesiones de captura de datos EEG.

Este experimento se divide en tres:

- **Experimentos con datos sintéticos :**

  Al tener mas datos en los conjuntos de datos, se tiene mas datos para realizar los entrenamientos durante mas tiempo sin llegar rápidamente a resultados que tiendan al al sobreajuste.

- **Experimentos con datos sintéticos y ventanas deslizantes :**

  Uniendo ventanas temporales al experimento aun se generan mas datos con los que realizar los entrenamientos.

- **Experimentos con datos sintéticos pero partición test con datos reales:**

  En este experimento se han creado datos sintéticos a partir de los conjuntos de datos segmentos 2, 3 y 4, y con estos datos se ha generado el subconjunto Train y Val. Para el subconjunto de datos Test se han utilizado los datos del segmento 1.

Herramientas de desarrollo

Las herramientas utilizadas son las siguientes:

- **Python y Bibliotecas:**

  Python:

  Seleccionado por su versatilidad y la amplia disponibilidad de bibliotecas especializadas en machine learning, redes neuronales y procesamiento de datos.

  Scikit-learn: Para algoritmos de machine learning tradicionales y técnicas de validación cruzada.

  TensorFlow y Keras: Para el desarrollo e implementación de redes neuronales avanzadas.

  Numpy y Scipy:

  Utilizado para manipulación y calculo numérico

Pandas:

Para manipulación y preprocesamiento de datos.

Matplotlib, Seaborn e Ipywidgets:

Para visualización de datos.

- **Entornos de Desarrollo:**

  Para el entorno de desarrollo del proyecto he utilizado Jupyter Notebooks que es una herramienta interactiva ampliamente utilizada en el análisis de datos, machine learning, redes neuronales.

  Permite combinar código, texto, visualizaciones y ecuaciones en un solo documento, lo que facilita la experimentación y documentación simultánea.

  Su capacidad de ejecutar celdas de código de manera independiente permite un desarrollo iterativo y rápido. Además, soporta múltiples lenguajes de programación, pero el más usado es Python.

  Las bibliotecas como pandas, numpy, matplotlib y seaborn se integran perfectamente, mejorando la eficiencia en la manipulación y visualización de datos.

- **Almacenamiento de datos:**

  Se ha llevado a cabo el almacenamiento de datos durante las ejecuciones de los modelos en Dataframes.

  Para almacenar y recuperar los datos de los resultados o de los Dataframes se han utilizado exportaciones e importaciones a archivos con formato CSV

5. Aspectos relevantes del desarrollo del proyecto

El desarrollo de este proyecto siguió la metodología CRISP-DM (Cross Industry Standard Process for Data Mining) [9], un enfoque estructurado y bien establecido para proyectos de minería de datos y análisis predictivo.

A continuación, se detallan los aspectos más interesantes y relevantes de cada fase del ciclo de vida del proyecto según CRISP-DM.

1. **Comprensión del Negocio**

   El objetivo principal fue identificar los objetivos y requisitos del proyecto desde una perspectiva empresarial.

   Se llevaron a cabo reuniones con los tutores para entender los objetivos específicos del proyecto, como la detección y clasificación de acciones de movimiento (arriba, abajo, derecha, izquierda).

   Se establecieron metas claras y medibles para el sistema de análisis de señales EEG, incluyendo los modelos a utilizar.

   En esta fase se tomo la decisión de crear un modelo modular con un Notebook Principal que realizara llamadas al resto de Notebooks secundarios siguiendo los objetivos de negocio de poder ser escalables para futuras investigaciones.

2. **Comprensión de los Datos**

   Unos de los objetivos principales fue familiarizarme con los datos disponibles y realizar un análisis preliminar.

Hubo una exploración inicial donde se exploro los conjuntos de datos EEG para comprender su estructura, características y distribución.

Se hizo uso de herramientas y visualizaciones para identificar patrones y posibles anomalías en los datos.

Se identificaron valores atípicos (outliners) que podrían afectar negativamente el rendimiento de los modelos.

Se realizaron cursos online básicos propuestos en las reuniones de seguimiento: Tutorial de Inicio de Pandas [**?**], Visualización de Datos [**?**], Trabajo con series temporales. [**?**]

3. **Preparación de los Datos**

   El objetivo principal de esta fase es preprocesar y preparar los datos para su análisis en los diferentes modelos.

   En esta fase se crearon nuevos conjuntos de datos para luego realizar análisis sobre ellos. En total 6 como se ha descrito con anterioridad.

   Se aplicación de técnicas de normalización y escalado, como StandardScaler y Z-Score, para estandarizar las señales EEG.

   Creación y utilización de ventanas deslizantes, agrupándolas por cada uno de los tipos de datos de cada conjunto de datos generado.

   Al final de la fase se propuso la generación de Datos Sintéticos para aumentar la cantidad y diversidad del conjunto de datos y abordar el análisis de los modelos desde otra perspectiva.

4. **Fase de Modelado**

   El objetivo en esta fase fue seleccionar y aplicar técnicas de modelado adecuadas. Los que se han utilizado para el proyecto son los siguientes:

   - **Algoritmos de Machine Learning:**

     K-Nearest Neighbors (KNN) Árboles de Decisión Random Forest

     Evaluación y Comparación: Uso de métricas como Tasa de acierto y loss para evaluar el rendimiento de estos modelos.

     Redes Neuronales: Multilayer Perceptron (MLP) Redes Neuronales Recurrentes (RNN) Long Short-Term Memory (LSTM)

     Optimización y Regularización:

     Uso de técnicas como Dropout para mejorar la generalización de los modelos.

Las dificultades que se encontraron en esta etapa fueron:

- Compilación de modelos de redes neuronales.

Al ejecutar los primeros modelos de redes neuronales los datos normalizados me proporcionaban errores de compilación con modelos SRNN o LSTM, tuve que cambiar el escalado de los datos datos y shapear los modelos para que pudieran ejecutarse.

- Utilización de ventanas temporales en los modelos de redes neuronales.

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, esto hizo que tuviera un gran retraso en la finalización y aceptación del codigo.

- Utilización de modelos de redes neuronales y callbacks.

El sobreajuste en los modelos predictivos era una constante y se implemento el uso de callbacks para evitar este sobreajuste.

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

5. **Evaluación**

   En esta fase se trata de poder evaluar el modelo para asegurar que cumple con los objetivos del negocio. Para ellos se utilizaran las siguientes técnicas:

   - **Métricas de Evaluación:**
     Para todos los modelos se ha utilizado la evaluación mediante subconjuntos de datos de validación y prueba. (Val y Test)
   - **Utilización de Callbacks y Técnicas de Monitorización:**
     Se ha hecho uso de EarlyStopping, ReduceLROnPlateau y ModelCheckpoint para evitar el sobreajuste y guardar el mejor modelo durante el entrenamiento.

6. **Despliegue**

   En esta fase el objetivo es la implementacion del proyecto y para ello se empezó con la definición y creación de los primeros Jupyter Notebooks.

   Jupyter Notebooks.

   Se han utilizado no solo para el desarrollo iterativo y la experimentación rápida, sino también para la implementación modular del proyecto.

La capacidad de crear, documentar y ejecutar celdas de código de manera interactiva permite a los usuarios finales poder ajustar y probar el proyecto en tiempo real, facilitando la depuración y el ajuste fino.

La naturaleza modular de los Jupyter Notebooks ha permitido separar distintas fases del proceso, desde la carga y preprocesamiento de datos hasta el entrenamiento y evaluación del modelo.

Esta separación ha facilitado el mantenimiento y la actualización de cada componente sin afectar al resto del sistema.

6. Trabajos relacionados

El análisis de las señales EEG (Electroencefalografía) es un campo de investigación extenso y dinámico con numerosas aplicaciones en la neurociencia, la medicina y la interfaz BCI.

A pesar del considerable progreso en esta área, aún existen desafíos significativos que impiden alcanzar tasas de acierto consistentemente altas y lograr una relevancia práctica más amplia.

Artículos científicos

- **Time Series Classification of Electroencephalography Data**

  En este articulo [1] de 2023, se escribe sobre varios experimentos realizados con señales EEG, en concreto en la pagina 606, el experimento llamado **Hand Movement Direction**, se basa en que dos individuos movieran un joystick hacia arriba, abajo, izquierda o derecha según su elección.

  Este articulo es el que mas se asemeja al proyecto actual, dando resultado de tase de acierto de como máximo 46,9 por ciento. Estos datos no son nada fructíferos al igual que el proyecto presentado.

- **EEG source imaging of hand movement-related areas: an evaluation of the reconstruction and classification accuracy with optimized channels**

  En este articulo [6] de Mayo de 2024 se detalla un experimento de clasificación de señales de dirección. A los individuos se les mostraba una flecha de dirección en una pantalla y ellos debían de pensar en esa dirección, se recogieron las señales EGG por interfaz BCI. El gran

trabajo de preprocesado de las señales de EGG han dado resultados altos pero no óptimos.

Este experimento no es igual al proyecto que se presenta pero si representa el variado grupo de investigaciones que se están realizando en la actualidad sobre señales EEG.

La mayor tasa de acierto para este articulo ha sido de 84,4 por ciento.

7. Conclusiones y Líneas de trabajo futuras

Conclusiones

- **Impacto de la Cantidad de Datos:**

  Los resultados obtenidos muestran que la falta de datos puede ser una limitación significativa en la obtención de altas tasas de acierto en el análisis de señales EEG. La disponibilidad de conjuntos de datos más grandes y variados podría mejorar la capacidad de generalización de los modelos y, por lo tanto, aumentar la precisión de las predicciones.

	KNN_TEST	TREE_TEST	RANDOM_TEST	MLP_TEST	RNN_TEST	LSTM_TEST
Segmento 1	0.692308	0.538462	0.538462	0.615385	0.615385	0.538462
Segmento 2	0.500000	0.375000	0.312500	0.375000	0.312500	0.250000
Segmento 3	0.400000	0.200000	0.200000	0.300000	0.350000	0.350000
Segmento 4	0.200000	0.300000	0.150000	0.300000	0.350000	0.350000
All Segmentos after	0.279412	0.338235	0.250000	0.367647	0.308824	0.338235
All Segmentos before	0.250000	0.308824	0.235294	0.308824	0.294118	0.323529

Figura 7.1: Ejemplo resultados Tasa de acierto primer experimento comparativo para subconjunto Test

Se puede ver a simple vista que los resultados para la métrica Tasa de acierto no han sido buenos para cualquiera de los modelos comparados.

Teniendo Tasas de acierto de hasta el 69,23 en uno de los segmentos individuales y de 36,76 en uno de los conjunto de datos completos.

- **Impacto del aumento de datos:**

  La inclusión de datos sintéticos ha demostrado ser beneficioso para mejorar las tasas de acierto en comparación con el uso exclusivo de datos reales.

  Este enfoque ha permitido aumentar la diversidad y la cantidad de datos disponibles para el entrenamiento, lo cual es crucial cuando los datos reales son limitados.

	RNN_RS_TEST	LSTM_RS_TEST
All Segmentos after	0.313333	0.463333
All Segmentos before	0.290000	0.346667

Figura 7.2: Ejemplo resultados Tasa de acierto experimento con aumento de datos a Test

  Estos resultados en comparación con los anteriores son un 10 por ciento mayores pero aun no son buenos resultados para la métrica Tasa de acierto.

- **Impacto de las ventanas temporales en los análisis:**

  Los datos recogidos tras utilizar las ventanas temporales con datos reales y con el aumento de datos sintéticos han sido los siguientes:

	RNN_SW_TEST	LSTM_SW_TEST
All Segmentos after	0.222222	0.222222
All Segmentos before	0.277778	0.222222

Figura 7.3: Ejemplo resultados Tasa de acierto experimento ventanas temporales a Test

	RNN_RS_SW_TEST	LSTM_RS_SW_TEST
All Segmentos after	0.627273	0.836364
All Segmentos before	0.545455	0.609091

Figura 7.4: Ejemplo resultados Tasa de acierto experimento ventanas temporales con aumento de datos a Test

El impacto combinado de las dos tecnicas para el analisis hacer que el aumneto en el porcentaje de la tasa de acierto, para el conjunto de datos estandarizado por segmentos y luego unificado en un solo conjunto de datos, alcanza has el 83.64 por ciento.

- **Validación de datos Sintéticos contra datos Reales:**

  La comparación directa entre conjuntos con aumento de datos y subconjuntos Test de exclusivamente datos reales ha llegado a mostrar que los resultados pueden llegar a superar significativamente el rendimiento de Tasa de acierto que de los que solo eran datos reales.

  Esto sugiere que los datos sintéticos pueden capturar mejor la variabilidad inherente en las señales EEG y mejorar la capacidad de generalización de los modelos.

  Unificando de nuevo las tecnicas de aumento de datos y ventanas deslizantes se ha llegado hasta el 64,17 por ciento en Tasa de Acierto, algo esperanzador para futuras lineas de investigación puesto que mejora en un casi 30 por ciento de Tasa de acierto con los modelos comparativos con solo datos reales.

	RNN_RS_TEST	LSTM_RS_TEST	RNN_RS_SW_TEST	LSTM_RS_SW_TEST
Test a datos reales	0.446154	0.484615	0.525000	0.641667

Figura 7.5: Ejemplo resultados Tasa de acierto experimento aumento de datos a datos reales Test

Líneas de trabajo futuras

- **Ampliación del Conjunto de Datos:**

  Una de las lineas de trabajo futuras seria la de seguir ampliando el conjunto de datos con mas individuos voluntarios que colaboraran en realizar este tipo de experimentos para así poder formar un conjunto de datos mas extenso y representativo.

- **Exploración de Datos Sintéticos y Ventanas Temporales:**

  La generación y uso de datos sintéticos ha mostrado ser efectiva en mejorar la Tasas de Acierto en los experimentos.

  Seria una buena linea de investigación futura. Se podría explorar y refinar técnicas de generación de datos sintéticos y segmentación

temporal con ventanas deslizantes para capturar mejor la variabilidad y las relaciones temporales en las señales EEG.

- **Implementación de Análisis de Componentes Principales (PCA):**

  La implementación de PCA puede proporcionar una visión más profunda de las características más relevantes en las señales EEG, facilitando así la interpretación y el análisis de los datos. Esto podría llevar a modelos más eficientes y comprensibles.

Bibliografía

[1] Rushbrooke Aiden Tsigarides Jordan Sami Saber Bagnall Anthony. Time series classification of electroencephalography data. https://link.springer.com/chapter/10.1007/978-3-031-43085-5_48", 2023.

[2] José Luis Pérez Gómez. app.diagrams.net. https://app.diagrams.net/. [Web en la que he creado los dibujos manualmente].

[3] José Luis Pérez Gómez. online.visual-paradigm.com. https://online.visual-paradigm.com/. [Web en la que he creado los dibujos manualmente].

[4] Dr. Sajil C. K. K-nearest neighbors algorithm. https://intuitivetutorial.com/2023/04/07/k-nearest-neighbors-algorithm/. [Internet; descargado 07-julio-2024].

[5] Ben Alex Keen. Feature scaling with scikit-learn. https://benalexkeen.com/feature-scaling-with-scikit-learn/.

[6] Soler A Giraldo E Molinas M. Eeg source imaging of hand movement-related areas: an evaluation of the reconstruction and classification accuracy with optimized channels. https://braininformatics.springeropen.com/articles/10.1186/s40708-024-00224-z", 2024.

[7] pixabay. Pixabay — increíbles imágenes gratis para descargar. https://pixabay.com/es/photos/eeg-integraci%C3%B3n-hirnstrommessung-2680957/. [Internet; descargado 06-julio-2024].

[8] Wikipedia. 68–95–99.7 rule — wikipedia, la enciclopedia libre. https://en.wikipedia.org/wiki/68%E2%80%9395%E2%80%9399.7_rule.

[9] Wikipedia. Cross industry standard process for data mining. https://es.wikipedia.org/wiki/Cross_Industry_Standard_Process_for_Data_Mining.