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 — 7 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, 7 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 algoritmos de procesado de señales y de machine/deep learning para la clasificación automática de señales.

Los datos aportados son de tipo EEG (Electroencefalograma) para la realización del TFG son datos referentes a experimentos basados en acciones 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.

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 subsiguiente.

- **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 como ampliar el conjunto de datos con la generación de nuevos datos sintéticos o la utilización de ventanas deslizantes 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 trastornos neurológicos basados en análisis de 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 Contexto Clínico o de Investigación:**

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

3. Conceptos teóricos

El proyecto se enfoca en el análisis y clasificación de señales EEG (Electroencefalograma) para detectar intenciones de movimiento mediante técnicas avanzadas de machine learning y redes neuronales.

3.1. 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.

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 EEG. [4]

Las principales señales 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.

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.

  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.

- **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 de que valores son outliners se ha utilizado la regla empírica o regla 68-95-99.7. [5]

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.

- **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.

- **Segmentación de Datos:** Se ha realizado segmentación de datos para dividir las señales EGG en ventanas temporales mas pequeñas asi se pueden identificar los experimentos a diferentes individuos realizados en el conjunto de datos.

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 vistos con anterioridad en la ejecución 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

- **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 sera 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 con anterioridad, lo cual es crucial para determinar si el modelo es apto para su implementación en un entorno real.

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 precisión 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 redes neuronales y los valores recomendados oscilan entre el 20 y 50 por ciento de neuronas desconectadas para que el aprendizaje no tienda al sobreajuste. Para este proyecto utilizare como valor, 30 por ciento.

- **Aumento de datos:** Generar nuevas muestras de datos 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 call back pero para el proyecto se utilizaran los siguientes:

  - **EarlyStopping:**
    Detiene el entrenamiento cuando una métrica monitoreada deja de mejorar. En los modelos que se han implementado se utilizara la métrica **val-loss** (La pérdida (loss) 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.
  - **ModelCheckpoint:**
    Guarda el mejor rendimiento que ha tenido durante el entrenamiento, basado en la métrica **val-loss** y se guarda en un archivo .keras.

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.

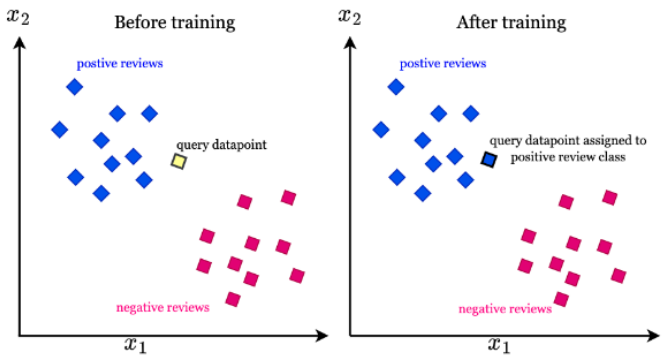

Figura 3.2: Uso KNN. [3]

  - **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.

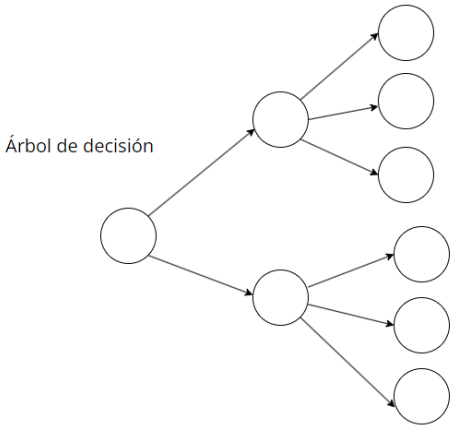

Figura 3.3: Uso arboles de decisión. [2]

- **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.

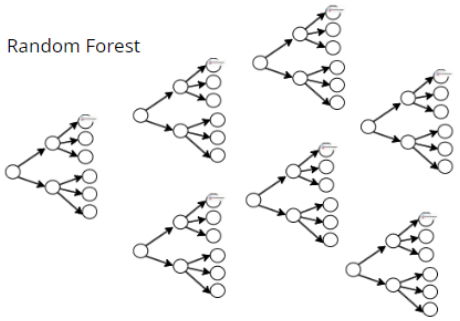

Figura 3.4: Uso random forest. [2]

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.

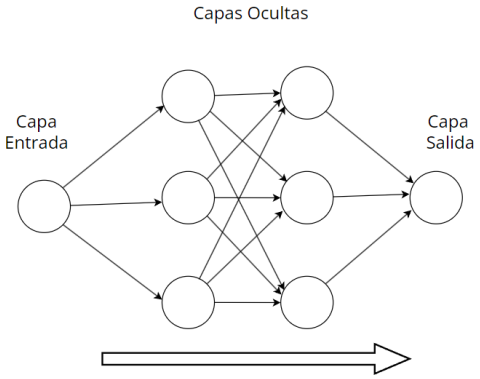

Figura 3.5: Red neuronal MLP. [2]

- **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.

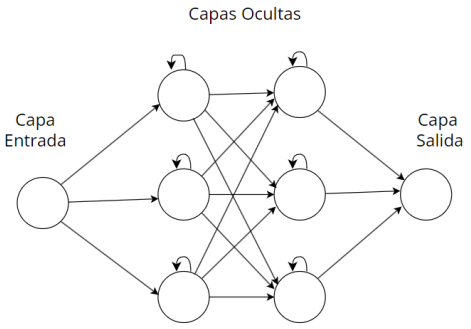

Figura 3.6: Red neuronal RNN. [2]

- **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 purstas 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.

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 3.7: Ejemplo ventanas temporales sobre targets. [1]

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.

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.

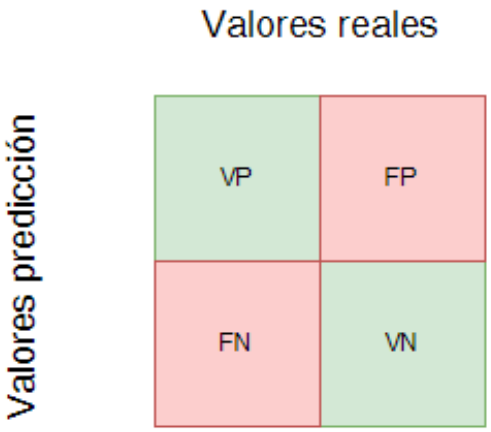

Figura 3.8: Matriz de confusión. [1]

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

Esta parte de la memoria tiene como objetivo presentar las técnicas metodológicas y las herramientas de desarrollo que se han utilizado para llevar a cabo el proyecto. Si se han estudiado diferentes alternativas de metodologías, herramientas, bibliotecas se puede hacer un resumen de los aspectos más destacados de cada alternativa, incluyendo comparativas entre las distintas opciones y una justificación de las elecciones realizadas. No se pretende que este apartado se convierta en un capítulo de un libro dedicado a cada una de las alternativas, sino comentar los aspectos más destacados de cada opción, con un repaso somero a los fundamentos esenciales y referencias bibliográficas para que el lector pueda ampliar su conocimiento sobre el tema.

5. Aspectos relevantes del desarrollo del proyecto

Este apartado pretende recoger los aspectos más interesantes del desarrollo del proyecto, comentados por los autores del mismo. Debe incluir desde la exposición del ciclo de vida utilizado, hasta los detalles de mayor relevancia de las fases de análisis, diseño e implementación. Se busca que no sea una mera operación de copiar y pegar diagramas y extractos del código fuente, sino que realmente se justifiquen los caminos de solución que se han tomado, especialmente aquellos que no sean triviales. Puede ser el lugar más adecuado para documentar los aspectos más interesantes del diseño y de la implementación, con un mayor hincapié en aspectos tales como el tipo de arquitectura elegido, los índices de las tablas de la base de datos, normalización y desnormalización, distribución en ficheros3, reglas de negocio dentro de las bases de datos (EDVHV GH GDWRV DFWLYDV), aspectos de desarrollo relacionados con el WWW... Este apartado, debe convertirse en el resumen de la experiencia práctica del proyecto, y por sí mismo justifica que la memoria se convierta en un documento útil, fuente de referencia para los autores, los tutores y futuros alumnos.

6. Trabajos relacionados

Este apartado sería parecido a un estado del arte de una tesis o tesina. En un trabajo final grado no parece obligada su presencia, aunque se puede dejar a juicio del tutor el incluir un pequeño resumen comentado de los trabajos y proyectos ya realizados en el campo del proyecto en curso.

7. Conclusiones y Líneas de trabajo futuras

Todo proyecto debe incluir las conclusiones que se derivan de su desarrollo. Éstas pueden ser de diferente índole, dependiendo de la tipología del proyecto, pero normalmente van a estar presentes un conjunto de conclusiones relacionadas con los resultados del proyecto y un conjunto de conclusiones técnicas. Además, resulta muy útil realizar un informe crítico indicando cómo se puede mejorar el proyecto, o cómo se puede continuar trabajando en la línea del proyecto realizado.

Bibliografía

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

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

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

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

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