UNIVERSIDAD DE ANTIOQUIA
1803

Seguimiento 4: Machine Learning

Danna Isabella García Saenz - Solanlly Evenedy Montoya Rivera

Mayo 31, 2024

Introducción

En el presente informe se presenta el análisis y desarrollo de un sistema de clasificación de gestos utilizando machine learning y un conjunto de señales electromiográficas (EMG), dónde se registraron patrones a partir de una pulsera MYO Thalmic ubicada en el antebrazo de diferentes usuarios y un PC con un receptor Bluetooth. Este proyecto se realiza teniendo en cuenta trabajos previos donde se exploraron diferentes aspectos relacionados con la fisiología de la señal EMG, su procesamiento, la extracción de características y el análisis de las mismas.

En la primera parte de este proyecto, se realizaron correcciones al primer trabajo entregado, optimizando el preprocesamiento y análisis de las señales EMG, en donde posteriormente, se definió una estrategia de filtrado de las señales utilizando filtros digitales, específicamente filtros IIR y FIR. La elección de la estrategia de filtrado se realizó teniendo en cuenta las capacidades del filtro FIR para mantener la integridad temporal de la señal, su estabilidad inherente y la flexibilidad en el diseño. Se procedió a obtener los índices de las señales filtradas aplicando el operador de energía de Teager-Kaiser (TKEO) y se compararon estos índices con los obtenidos en el proyecto inicial. Este análisis permitió evaluar la eficacia del filtrado y su impacto en la calidad de la señal para la extracción de características relevantes.

Después de analizar los resultados obtenidos, se decidió la mejor estrategia para continuar con el desarrollo del modelo de clasificación, en donde se implementaron y evaluaron tres diferentes arquitecturas de redes neuronales, analizando sus resultados mediante matrices de confusión y validación cruzada, con el objetivo de determinar la arquitectura más adecuada para la clasificación de gestos a partir de las señales EMG.

Por último, se exploraron y se implementaron otros modelos de machine learning para que de esta manera se pudiera comparar el desempeño de los otros modelos con el de las redes neuronales y determinar su viabilidad para la tarea de clasificación de gestos. Como parte de la exploración de otros modelos, también se discutieron las posibilidades de implementar estas soluciones en sistemas embebidos, seleccionando el tipo de microcontrolador más adecuado y analizando los tiempos de adquisición y clasificación de las señales, con el objetivo de desarrollar un sistema práctico y eficiente para la clasificación de gestos en tiempo real.

b. Definir una estrategia de filtrado de las señales del proyecto 1 usando filtros IIR o FIR justificando la selección.

Al filtrar señales EMG la elección de un filtro FIR se justifica por varias razones, estos son conocidos por su estabilidad y facilidad de diseño en comparación con los filtros IIR lo cual es crucial en el procesamiento de señales biológicas donde la precisión y la fiabilidad son fundamentales para la interpretación correcta de los datos fisiológicos. Los filtros FIR son inherentemente estables, lo que significa que no presentan problemas de inestabilidad que a veces pueden ocurrir en los IIR, la estabilidad es importante en el procesamiento de señales EMG donde debe mantenerse una integridad de los datos para la interpretación de la actividad muscular, y además estos son lineales lo que garantiza que la superposición de señales se maneje de manera consistente, lo que es importante al tratar con señales que pueden contener múltiples componentes de interés, el tipo de señales de ruido que se filtran en el contexto de la EMG es comúnmente los ruidos de línea eléctrica, de movimiento muscular o interferencias electromagnéticas, el filtro FIR diseñado con una frecuencia de corte de 50 Hz se enfoca en atenuar

eficazmente el ruido de alta frecuencia, como el ruido de línea eléctrica, mientras preserva las características de interés de la señal EMG [1].

En el código que se implementó al utilizar este tipo de filtros tenemos un control directo sobre los coeficientes del filtro para una mayor flexibilidad para adaptarse a las características específicas de la señal EMG y del ruido presente en ella, en este caso el uso de la ventana de hamming en el diseño ayuda a minimizar el efecto de las discontinuidades en la respuesta en frecuencia, respecto a los parámetros que seleccionamos el número de coeficientes del filtro se elige como 101 en este caso para lograr un equilibrio entre la precisión del filtrado y la eficiencia computacional (si se escoge un número mayor puede proporcionar una mayor selectividad en la frecuencia de corte pero también aumentaría la complejidad computacional) además, la frecuencia de muestreo de 1000 Hz (fs) se considera adecuada para señales EMG ya que permite capturar con precisión las variaciones temporales de la señal y son valores que ya hemos utilizado en los proyectos anteriores.

d. Analizar y discutir los resultados del punto anterior de manera que para los puntos siguientes se usen, debidamente sustentado, los índices obtenidos con la señal filtrada o la señal sin filtrar.

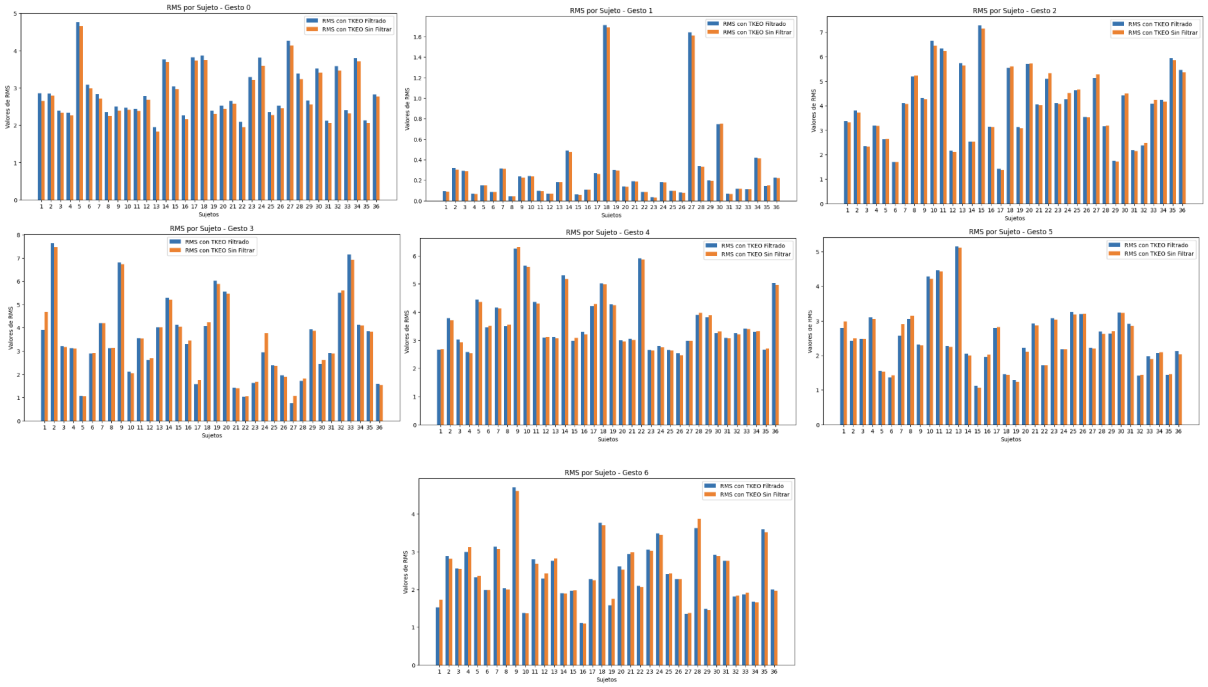

***Figura 1**. Algunos valores de RMS evaluados en todos los sujetos para la señal TKEO filtrada y TKEO sin filtrar*

En los resultados obtenidos, se observó que los valores de RMS (valor cuadrático medio) de la señal filtrada son consistentemente más altos que los valores de la señal original sin filtrar en todos los gestos y sujetos analizados, el comportamiento podría explicarse por razones como que aunque el filtro tiene como objetivo eliminar componentes de ruido también produce un efecto de suavizado en la señal, lo que reduce las fluctuaciones bruscas y picos agudos, además el filtrado puede alterar la distribución de energía en la señal concentrando una mayor proporción en las componentes de baja frecuencia y esto podría resultar en el aumento del valor cuadrático medio de la señal filtrada, es importante destacar que el incremento en los valores de RMS de la señal filtrada no implica necesariamente una amplificación de la señal EMG en sí misma, más bien podríamos decir que refleja la preservación de las componentes de baja frecuencia que son relevantes y correcta eliminación de ruido.

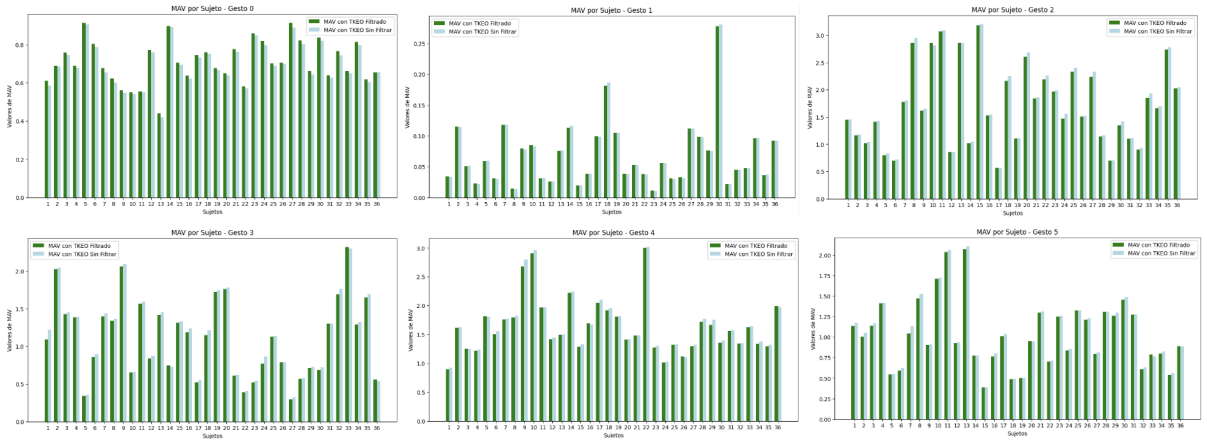

***Figura 2**. Algunos valores de MAV evaluados en todos los sujetos para la señal TKEO filtrada y TKEO sin filtrar*

En las gráficas de MAV (valor medio absoluto) se observa que los valores para la señal filtrada son generalmente más pequeños que los valores de la señal sin filtrar, esto se debe a que el filtrado suaviza la señal, reduciendo las fluctuaciones bruscas lo que disminuye la amplitud media absoluta de la señal filtrada, aparte de eso el filtro elimina el ruido que suele tener amplitudes significativas y aumentan el MAV de la señal sin filtrar y por lo mismo la señal filtrada tiene un MAV más bajo, los gráficos de caja y bigotes (boxplots) permiten comparar las distribuciones de las características y podemos observar una menor dispersión y un rango más estrecho en las distribuciones de las características para la señal filtrada, lo que indica una menor variabilidad y mayor confiabilidad en los valores obtenidos.

Si bien la señal sin filtrar puede retener más información original, el uso de la señal filtrada se considera una mejor opción para el análisis de señales, el filtrado FIR permite eliminar eficazmente el ruido y la estabilidad e integridad de los datos para el correcto análisis de la actividad muscular.

f. Consultar, justificar el uso e implementar otros modelos de machine learning que permitan la clasificación de gestos.

El **modelo de Random Forests** es una técnica de machine learning que se destaca por su capacidad para realizar tareas de clasificación con alta precisión y robustez, las ventajas clave del modelo incluyen su capacidad para manejar conjuntos de datos grandes con alta dimensionalidad, su resistencia al sobreajuste, y su capacidad para manejar tanto variables numéricas como categóricas, además, este modelo es relativamente fácil de entrenar y ajustar, lo que lo hace ideal para aplicaciones donde se requiere un rendimiento sólido sin una configuración compleja. En cuanto a los requisitos de datos, se necesita una cantidad suficiente de datos y deben ser de alta calidad, libres de ruido y con etiquetas precisas para garantizar un entrenamiento efectivo del modelo, este modelo en términos de recursos computacionales no requiere una potencia computacional extrema y puede ejecutarse eficientemente en un computador estándar, para implementar el modelo se pueden utilizar bibliotecas como scikit-learn en python, que ofrece una implementación eficiente y fácil de usar de este algoritmo, el proceso de preprocesamiento de datos puede incluir la normalización de características, eliminación de valores atípicos y codificación de variables categóricas, según sea necesario para optimizar el rendimiento del modelo.

Para la arquitectura del modelo no se tienen capas ni neuronas en el sentido tradicional de las redes neuronales, en cambio se compone de múltiples árboles de decisión que trabajan de forma conjunta para realizar la clasificación. Cada árbol se entrena de forma independiente en subconjuntos aleatorios de datos y los resultados individuales se combinan mediante un proceso de votación mayoritaria o promedio para obtener el resultado final del modelo, se podría decir que la técnica de ensamblaje de múltiples árboles de decisión proporciona una mayor precisión y robustez en comparación con un solo árbol de decisión y en el contexto de clasificación de gestos de la mano, este modelo podría ser adecuado debido a su capacidad para manejar múltiples características y patrones complejos presentes en los datos de gestos [2].

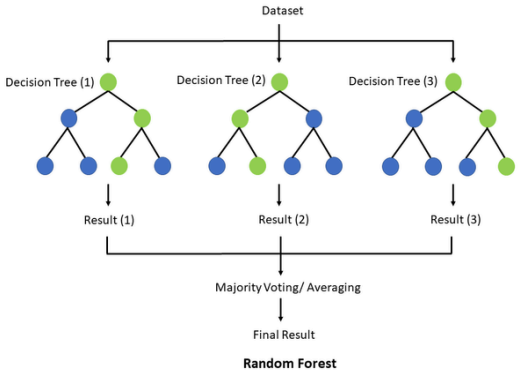

Figura 3. *Grafica funcionamiento Random Forest*

La **regresión logística** es un modelo ampliamente utilizado en el machine learning para predecir la probabilidad de que una variable dependiente pertenezca a una categoría específica, para este modelo se necesitan una cantidad suficiente de datos etiquetados que representen una variedad de gestos de la mano y la calidad de los datos es crucial para evitar sesgos y garantizar la precisión del modelo, para implementar el modelo, se pueden utilizar bibliotecas como scikit-learn en python y en cuanto al preprocesamiento de datos para el modelo podría incluir la normalización de características, la codificación de variables categóricas y la división del conjunto de datos en conjuntos de entrenamiento y prueba. En cuanto a la arquitectura del modelo, la regresión logística se basa en una función logística para estimar las probabilidades de pertenencia a una clase, la curva de regresión logística se ajusta a los datos de entrenamiento mediante la estimación de los coeficientes de regresión, lo que permite predecir la probabilidad de pertenencia a una clase en función de los valores de las variables independientes, la curva se obtiene a partir de la función logística, que transforma la salida de la regresión lineal en una probabilidad entre 0 y 1. Teniendo en cuenta que se utiliza comúnmente para problemas de clasificación binaria, donde la variable dependiente tiene dos categorías o clases distintas, por lo tanto en el contexto de clasificar gestos de la mano, la regresión logística no sería el modelo de Machine Learning más adecuado debido a la necesidad de para manejar variables categóricas más de dos gestos de mano, se debe de explorar variantes de la regresión logística o modelos de clasificación multiclase [3].

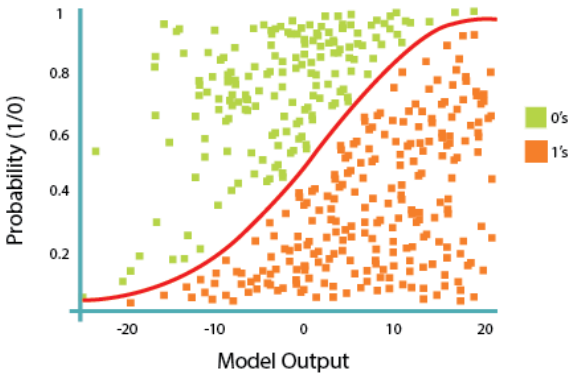

Figura 4. *Grafica funcionamiento Regresión logística*

El **modelo de k-NN (k-Nearest Neighbors)** es un algoritmo de machine learning que se utiliza para clasificar nuevas muestras basándose en la similitud con las muestras de entrenamiento, cuando se recibe una nueva muestra, el modelo compara esta muestra con todas las muestras existentes y selecciona las 'k' muestras más cercanas, luego asigna la clase mayoritaria de estas 'k' muestras a la nueva muestra, este modelo es de aprendizaje basado en instancias ya que no hay una fase de entrenamiento explícita sino que almacena todas las instancias de entrenamiento y realiza la clasificación en tiempo de consulta, sus ventajas serian la capacidad para trabajar con datos numéricos y nominales sin hacer suposiciones sobre la distribución de los datos, sin embargo hay que tener en

cuenta que tiene un alto costo computacional debido a la necesidad de calcular las distancias para cada nueva muestra y su gran requerimiento de memoria para almacenar todas las muestras de entrenamiento, para el modelo la cantidad de datos necesaria depende del problema específico, pero generalmente se requieren muchas muestras, es crucial que los datos de entrenamiento y de prueba provengan de la misma distribución.

Para implementar k-NN, se pueden utilizar bibliotecas como NumPy para cálculos numéricos, SciPy para funciones científicas y estadísticas, y scikit-learn, que proporciona una implementación eficiente y fácil de usar del algoritmo k-NN, es importante manejar los datos faltantes y limpiar cualquier ruido que pueda afectar la precisión del modelo, la arquitectura del modelo se basa en el cálculo de distancias las funciones de distancia más comunes utilizadas son la distancia euclidiana, manhattan y de Minkowski, se selecciona la más adecuada según las características específicas del problema, este modelo podría ser adecuado para estudios de clasificación de gestos de la mano debido a su facilidad de implementación [4].

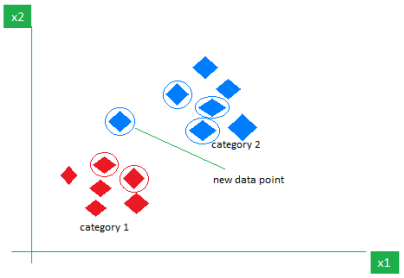

Figura 5. *Grafica funcionamiento k-Nearest Neighbors*

El **TensorFlow** es una librería de código libre para Machine Learning (ML), esta librería permite construir y entrenar redes neuronales para detectar patrones y razonamientos usados por los humanos. TensorFlow es una herramienta poderosa y versátil que permite a los desarrolladores e investigadores construir y desplegar modelos avanzados de aprendizaje automático y profundo en una variedad de plataformas y dispositivos. Su flexibilidad, rendimiento y amplio ecosistema lo convierten en una opción ideal para una amplia gama de aplicaciones en el campo de la inteligencia artificial [5][6].

Las aplicaciones de TensorFlow son extremadamente versátiles y se pueden ejecutar en una amplia variedad de entornos, incluyendo máquinas locales, clústeres en la nube, dispositivos iOS y Android, así como en CPUs y GPUs. Si se opta por utilizar la infraestructura en la nube de Google, TensorFlow también puede aprovechar las Tensor Processing Units (TPUs) de Google para obtener una mayor velocidad de procesamiento. Además, los modelos desarrollados con TensorFlow pueden ser implementados en diversos dispositivos para llevar a cabo predicciones de manera eficiente, en la figura 6 se pueden detallar algunas de las ventajas que se tienen con TensorFlow como ya se había mencionado anteriormente [6].

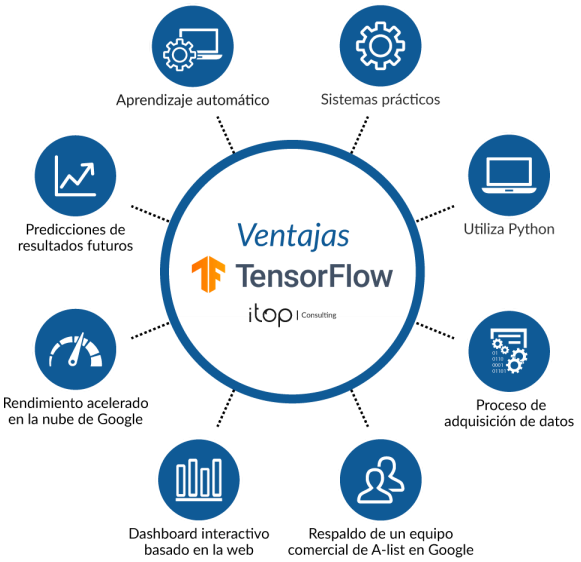

Figura 6. *Gráfica las ventajas del TensorFlow [5]*

g. Discutir cómo se podría implementar la solución del punto e y f usando sistemas embebidos seleccionando el tipo de microcontrolador y los posibles tiempos de adquisición y clasificación de las señales

Para el caso del microcontrolador seleccionado debe tener suficiente potencia de procesamiento para ejecutar el algoritmo de clasificación de gestos en tiempo real, algunos podrían ser ARM Cortex-M series (ofrece un buen rendimiento con memoria RAM y Flash, interfaces de comunicación), también microcontroladores de señal digital (DSP) (optimizados para el procesamiento de señales y pueden ser una opción adecuada para aplicaciones que involucran señales EMG).

Para adquirir las señales EMG, se necesitará un circuito de acondicionamiento de señal que amplifica y filtra las señales antes de digitalizarlas. el circuito puede ser externo o interno en el microcontrolador si cuenta con conversores analógicos ADC, el tiempo de adquisición de las señales EMG dependerá de la frecuencia de muestreo requerida que seria de 1000 Hz para capturar adecuadamente las características de las señales EMG, lo que implica un tiempo de adquisición de 1 milisegundo por muestra, la fase de procesamiento y clasificación de las señales implican varios pasos, como el filtrado, la extracción de características y la clasificación utilizando el modelo de aprendizaje automático entrenado, es importante tener en cuenta que el procesamiento y la clasificación de señales en tiempo real implican restricciones de tiempo estrictas. Por lo tanto, es crucial optimizar el código y seleccionar cuidadosamente los algoritmos y las técnicas de procesamiento de señales para garantizar un rendimiento adecuado.

LINK DEL VIDEO
https://youtu.be/qHVkosW0M0M?feature=shared

Conclusiones

- Se implementaron las correcciones sugeridas en el Proyecto 1, mejorando la precisión del procesamiento y la interpretación de las señales EMG. Los ajustes incluyeron la refinación de los métodos de adquisición de datos, la optimización de los algoritmos de extracción de características y el filtrado de las señales, en donde la selección de filtros FIR se justificó por su estabilidad y linealidad de fase, características críticas para mantener la integridad de la señal EMG durante el procesamiento. Como resultado de la comparación entre los índices de la señal filtrada y la señal sin filtrar reveló que la filtración contribuye significativamente a la reducción del ruido y a la mejora de

la claridad de los datos relevantes. Esto justifica el uso de índices derivados de la señal filtrada para los análisis subsecuentes, garantizando resultados más robustos y fiables.

- En el análisis de las arquitecturas propuestas para las redes neuronales, se observaron diferencias significativas en los valores de precisión obtenidos. La primera arquitectura y la tercera arquitectura presentaron valores de precisión que oscilaban entre el 60% y el 62%. Aunque los datos de precisión (accuracy) en el conjunto de entrenamiento de la primera arquitectura fueron superiores a los de la tercera, la diferencia no fue sustancial. La primera arquitectura consistía en 2 capas, cada una con 9 neuronas. Por otro lado, la tercera arquitectura tenía 3 capas con 20 neuronas cada una. Esta diferencia en la configuración de las capas y neuronas afectó los resultados de precisión y rendimiento de las redes.
- Además de las redes neuronales, se investigaron otros modelos de machine learning como Support Vector Machines (SVM), k-Nearest Neighbors (KNN) y Random Forest. La justificación de su uso se basó en la simplicidad y eficiencia computacional, ofreciendo un buen equilibrio entre precisión y tiempo de procesamiento para la clasificación de gestos. La implementación del primer y último modelo demostró que pueden ser alternativas viables a las redes neuronales, especialmente en escenarios con recursos computacionales limitados con resultados de precisión muy buenos en el caso del modelo de TensorFlow.

Bibliografía

[1] Hemant Kumar and Anjana Goen. "Comparative Study of FIR Digital Filter for Noise Elimination in EMG Signal." International Journal of Advanced Research, Volume 3, Issue 12, pp. 598-603 (2015). Obtenido de: https://www.researchgate.net/publication/330703731_Comparative_Study_of_FIR_Digital_Filter_for_Noise_Elimination_in_EMG_Signal

[2] Breiman, L. Random Forests. En Machine Learning (p. 587-604). Springer. (2001) Obtenido de: https://www.math.mcgill.ca/yyang/resources/doc/randomforest.pdf

[3] Marina Domínguez Martín. "Regresión Logística y Técnicas de Aprendizaje. Aplicaciones." Trabajo de Fin de Grado, Universidad de Zaragoza, (2021). pp. 1-58. Obtenido de: https://zaguan.unizar.es/record/110300/files/TAZ-TFG-2021-3094.pdf

[4] Rahul Vishwakarma, Jyothish Kumar J. KNN: K-nearest Neighbours. School of Computer Sciences, National Institute of Science Education and Research, Bhubaneshwar, Homi Bhabha National Institute. 2023. Obtenido de: https://www.niser.ac.in/~smishra/teach/cs460/23cs460/lectures/lec13.pdf

[5] TensorFlow. Itop Consulting. Obtenido de: https://www.itop.es/soluciones-tecnologicas/business-analytics-business-intelligence/tensorflow.htm

[6] TORRES, Jordi. Python deep learning: Introducción práctica con Keras y TensorFlow 2. Alpha Editorial, 2020. Obtenido de: https://books.google.es/books?hl=es&lr=&id=0XJ6EAAAQBAJ&oi=fnd&pg=PR10&dq=qu%C3%A9+es+tensorflow&ots=E3nSJuEDAU&sig=I2GoJ4k8vLwpEWdlQd1EVcSfFGM