TRABAJO FIN DE GRADO

PREDICCIÓN EVENTOS TROMBÓTICOS EN PACIENTES COVID-19 MEDIANTE REDES NEURONALES

TRABAJO FIN DE GRADO PARA LA OBTENCIÓN DEL TÍTULO DE GRADUADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

FEBRERO 2023

Eduardo Galeote Escalera

DIRECTOR DEL TRABAJO FIN DE GRADO:
Pablo Garrido Martínez-Llop

INDUSTRIALES
ETSII | UPM

POLITÉCNICA

"Que nadie se acerque jamás a ti sin que al irse se sienta un poco mejor y más feliz"

- Madre Teresa de Calcuta

AGRADECIMIENTOS

He querido dedicar unas palabras a las personas más importantes que han estado alrededor mio durante este último año. Son mi pilar de apoyo y también lo han sido en este TFG.

Mis amigos. Los grupos de PICASSO y ScanFC, del colegio y universidad respectivamente, me han ayudado sacándome una sonrisa en los necesitados tiempos de descanso.

A mi familia. Mi madre, mi padre y mi hermano. Hacen muy fácil el día a día con su inmensurable cariño y apoyo. Especial mención a mi padre. Es cardiólogo intervencionista en el Hospital La Paz y ha estado involucrado de gran manera, desde la idea inicial hasta la finalidad del proyecto.

Por supuesto agradecer a mi tutor Pablo Garrido. No sólo me ha guiado y tutorizado en este extenso trabajo, ha sido también un gran amigo. Es un gran profesional y lo más importante en esta vida, buena persona.

RESUMEN EJECUTIVO

La enfermedad COVID-19 es una enfermedad infecciosa causada por el virus SARS-CoV-2. La pandemia debida a este virus comenzó en 2019. Desde entonces ha habido un total de 7 millones de muertes. Se han estado probando diversos medicamentos y vacunas para evitar que este número aumente con la rapidez que lo ha hecho estos pocos años. La idea de este trabajo surge de aquí, de encontrar una manera de poder ayudar a resolver este problema al cual se enfrenta muchos países por todo el mundo.

Durante estos años se ha ido observando que los pacientes con infección por COVID-19 suelen tener un mayor riesgo de eventos trombóticos. Éstos incluyen trombosis venosas y arteriales. Los trombos pueden dar lugar a distintas enfermedades según el nivel al que afecten como ictus, infartos, isquemias, trombosis venosa profunda o tromboembolismo pulmonar. Se ha tratado de identificar los factores de riesgo de estos eventos en los pacientes con COVID-19, lo que podría ayudar a los médicos en la prevención, la identificación temprana y el manejo de la enfermedad en los pacientes con COVID-19 hospitalizados y mejorar los resultados clínicos en estos pacientes.

Por otro lado tenemos el Machine Learning. Se trata de una rama de la inteligencia artificial cuyo fin es el aprendizaje de las máquinas. Usamos la potencia de los ordenadores para identificar patrones entre millones de datos con el objetivo de hacer predicciones. Tiene multitud de aplicaciones, como en las aplicaciones informáticas que usas en el móvil o la detección de fraude en el uso de tarjetas de crédito. Las redes neuronales se basan en datos de entrenamiento para aprender y mejorar su precisión con el tiempo.

Nace el objetivo de este TFG, el cual es unir dos modalidades completamente distintas. La medicina y la inteligencia artificial. Se propuso aplicar la tecnología de las redes neuronales al COVID-19. Partiendo de una base de datos del Hospital Universitario La Paz. Trata de una base de datos producida por el servicio de cardiología en la que grabaron las casi 500 variables de cada paciente hospitalizado durante los tres primero meses. Este servicio publicó un *paper* con la ida de predecir los eventos trombóticos de los pacientes ingresados por COVID-19 mediante parámetros estadísticos.

A raíz de esta idea llegamos a la de este proyecto, conseguir encontrar una red neuronal, que entrenándola con la misma base de datos, consiga predecir con buena precisión el desarrollo de eventos tromboembólicos en pacientes hospitalizados.

El procedimiento de este trabajo ha consistido en una investigación sobre los trabajos ya hechos en la predicción de enfermedades con inteligencia artificial. También ha tenido lugar una formación para poder entender y replicar todas la información vista.

En lo práctico, se ha dedicado un largo tiempo a la limpieza de la base de datos. De la base de la que partíamos contenía muchos caracteres en blanco y variables con escasa información. Este tipo de datos no sirve para entrenar una red neuronal, por lo que una depuración y estudio estadístico se llevó a cabo. Más tarde comenzamos con el desarrollo de las redes neuronales. La cantidad de hiperparámetros a modificar en una red es inmensa.

La dedicación a la prueba y entrenamiento de estas redes ha sido considerable en el proyecto. Se han probado los distintos parámetros y estructuras para encontrar la óptima que consiga resolver nuestro problema con el menor error posible.

Encontramos una red neuronal con parámetros expuestos en la siguiente tabla que consigue predecir el desarrollo de eventos tromboembólicos con una precisión del 96.7 %. Obteniendo un

error cuadrático medio de 0.0308.

Parámetros	
Estructura	[38,26,1]
Nº Capas	3
F. Activación	ReLu + Sigmoid
F. Coste	MSE
Callbacks	Early Stopping
Optimizador	Adam

Tabla 0.1: Modelo red neuronal

Existen otros objetivos logrados. Se ha profundizado de una manera extensa en la inteligencia artificial, más en concreto, en las redes neuronales. También nos hemos informado de la diversos aspectos de la medicina y hemos hablado con expertos en la materia para aumentar el conocimiento para la posible realización del trabajo. Hemos aportado al estado del arte de la medicina con al Machine Learning y las posibilidades que este campo tiene.

Las líneas futuras de este trabajo no son pocas. Para empezar, en proyectos relacionados con redes neuronales siempre se pueden emplear una mejor base de datos con la que entrenar la red. Nuestro caso no es excepción, una base de datos más completa mejoraría el resultado y predicción de nuestro objetivo. Ya hablando de medicina en general, tiene diversas aplicaciones el *Deep Learning* que servirá de gran ayuda para los profesionales médicos del futuro y ayudarán a salvar muchas vidas. La posibilidad de analizar millones de datos históricos abre un mundo de posibilidades para todas las especialidades en la medicina, ya sean cardiología, neurología, dermatología, odontología, etc.

Palabras clave: Eventos trombóticos, redes neuronales, análisis de datos, predicción, COVID-19, medicina.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1. INTRODUCCIÓN

Vamos a realizar un estudio sobre eventos trombóticos y su predicción con el *Machine Learning*. Comentaremos ciertos aspectos del COVID-19 así como la historia de esta enfermedad y el uso e historia de la intelgiencia artificial que usaremos para nuestro objetivo.

1.1. Introducción al COVID-19

La enfermedad COVID-19 es una enfermedad infecciosa causada por el virus SARS-CoV-2. El primer caso de la infección se desarrolló el día 20 de diciembre de 2019, de ahí su denominación. En la localidad de Wuhan, China, se descubrió un brote epidémico que afectó a unas 60 personas. Éste afectó de manera grave a muy pocos. Así, días mas tarde ingresarían en un hospital del país 4 pacientes por síntomas de neumonía. Al no encontrar la causa de esta neumonía, se notificó el último día del año a la Organización Mundial de la Salud (OMS) lo que sería meses más tarde una pandemia.

A principios del 2020 se procedió a aislar el virus por parte de las autoridades chinas. Se hizo un estudio del genoma el cual fue enviado de inmediato a la OMS para la realización de diagnósticos vía pruebas PCR en distintos países.

Ciertos pacientes desarrollaron algunos síntomas más graves que otros, entre estos: fiebre, malestar, tos seca, dificultad para respirar y fallo respiratorio.

El virus fue expandiéndose rápido, dándose mas adelante los primeros casos en Tailandia y Japón.

A finales del mes de enero, habiéndose propagado por más de una decena de países, la OMS declaró una emergencia sanitaria de alcance internacional.

El 11 de marzo la infección se había diagnosticado en mas de 100 países, reconociéndose pandemia por la OMS. A finales del mes ya llevaban 500 mil casos diagnosticados.

En estas fechas empezó a cundir el pánico por todo el mundo, viendo como los gobiernos tomaban medidas drásticas para evitar la expansión del virus. Restricciones del máximo nivel como las que todos hemos experimentado. Desde tener que llevar una simple mascarilla en el metro hasta no poder salir de nuestras casas ni para dar una vuelta.

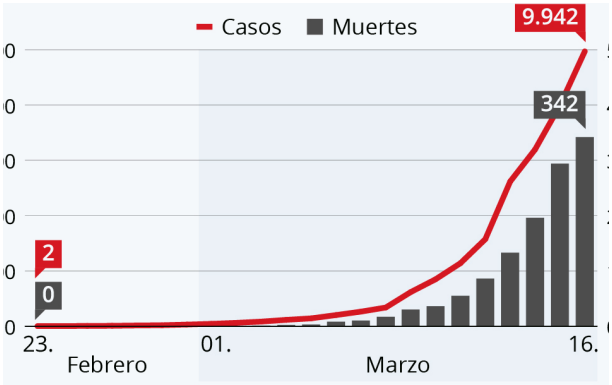

Figura 1.1: Evolución mortalidad COVID-19

Aunque el índice de mortalidad no fuese lo más grave comparado con otro tipo de enfermedades, la rápida expansión y elevado volumen de pacientes con síntomas graves sí lo fue. Los hospitales se vieron desbordados e incapaces de atender a gente, dándose así decenas de miles de muertes sólo en España.

Incluso en la actualidad, la pandemia COVID-19 sigue siendo un reto para sistemas sanitarios por todo el mundo. Son por estas razones lo que hace muy atractivo su estudio.

1.2. Introducción al Machine Learning

Por otro lado tenemos el *Machine Learning*. Se trata de una rama de la inteligencia artificial cuyo fin es el aprendizaje de las máquinas. Usamos la potencia de los ordenadores para identificar patrones entre millones de datos con el objetivo de hacer predicciones. Tiene multitud de aplicaciones, como en las aplicaciones informáticas que usas en el móvil o la detección de fraude en el uso de tarjetas de crédito. Son algoritmos matemáticos que usan principalmente la estadística para el funcionamiento de esta capacidad.

No fue hasta 1950 que distintos científicos empezaron investigar cómo aplicar la biología de las redes neuronales del cerebro humano. La idea conllevó la creación de las redes neuronales artificiales (RNA) que usaremos en este trabajo.

Éstas redes neuronales son un subconjunto del *Machine Learning* y son el núcleo de algoritmos de *deep learning* que explicaremos mas adelante. Su nombre y estructura se inspiran en el cerebro humano, imitando la forma en que las neuronas biológicas se comunican entre sí.

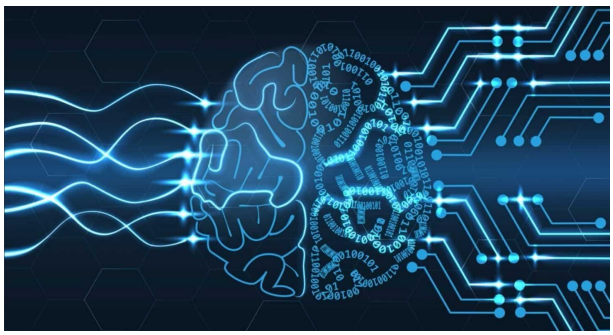

Figura 1.2: Aprendizaje automático

Las redes neuronales se basan en datos de entrenamiento para aprender y mejorar su precisión con el tiempo. Sin embargo, una vez que estos algoritmos de aprendizaje se afinan en cuanto a precisión, son poderosas herramientas en la informática y la inteligencia artificial, que nos permiten clasificar y agrupar datos a gran velocidad. Una de las redes neuronales más conocidas y que mas usamos es el algoritmo de búsqueda de Google.

Trataremos de combinar el COVID-19 con algoritmos de redes neuronales para hallar un modelo, que basado en una base de datos de pacientes COVID-19, nos genere una respuesta precisa sobre algunos campos de interés que mas tarde describiremos.

1.3. Motivación

El motivo principal es la atracción personal por el análisis de datos empleando modelos de inteligencia artifical. El aprendizaje de esta técnica supone una gran motivación. En búsqueda de una base de datos con la que trabajar, pareció bastante interesante usar el COVID-19 debido al impacto que nos ha supuesto y el gran interés por encontrar soluciones que ayuden a detener la pandemia.

El *Machine Learning* está en boca de muchos. No es sorpresa que se haya escuchado más de una vez en nuestra vida cotidiana. El amplio avance, progreso y demanda en este mercado supone una motivación importante. Sobre todo un estudio de sus distintas técnicas y diferencias.

Especialmente podemos ubicar muchas posibles ayudas del Machine Learning en el campo de la medicina. En primer lugar, en muchas especialidades existen una cantidad inmensa de datos que un ser humano tiene que poder analizar. En segundo lugar, la experiencia es el aspecto que más destaca en un buen médico. Una red neuronal nos permite introducir miles de millones de datos con todas las historias clínicas disponibles de pacientes y producir un resultado muy similar que pueda servir de ayuda al profesional. Juntando la cantidad de variables de un ser humano y la experiencia que puede adquirir una inteligencia artifical al introducir los datos históricos de las enfermedades, obtenemos modelos de redes neuronales que puedan ayudar significativamente a la toma de decisiones, a veces muy complicada, en este campo.

Existen numerosos síntomas que resultan atractivos de investigar mediante redes neuronales. Uno de ellos es la aparición de evento trombóticos en pacientes ingresados que han dado positivo en la prueba PCR. Una predicción de si un paciente puede o no desarrollar éste tipo de síntomas puede ayudar a salvar muchas vidas.

Por lo mencionado anteriormente realizaremos una investigación del desarrollo de trombos en pacientes COVID-19 mediante el uso de redes neuronales artificiales.

2. ESTADO DEL ARTE

Dado el impacto masivo de la pandemia por el COVID-19 y el desastre a nivel global que ha causado, existen numerosos estudios dedicado a esta temática. Debido a la rápida expansión y las graves consecuencias del virus, muchos científicos empezaron a investigar y a intentar hallar una vacuna para tratar de poner un punto y final a esta tragedia.

Gracias a esta urgencia, a día de hoy disponemos de varias vacunas muy eficaces. Respecto a los estudios de las bases de datos recolectadas, existen numerosos estudios relacionados con el COVID-19. En total unos 730 mil *papers* dedicados a comentar cada aspecto relevante de este virus y su afectación a nivel social, económico y biológico.

En la *World Health Organization* se puede hacer una búsqueda de miles de investigaciones. Si buscamos por *"prediction"* nos encontramos con 14 mil resultados. Modelos creados por todo el mundo basándose en inteligencia artificial y distintos métodos.

Este trabajo estará basado en un estudio realizado por médicos del Hospital La Paz en Madrid. El grupo de cardiólogos se apoyo en un abase de datos propia, de pacientes ingresados en el hospital, para su estudio.

2.1. Machine Learning y medicina

El aprendizaje automático es un campo en rápido crecimiento con numerosas aplicaciones en varias industrias, incluida la medicina. En los últimos años, ha habido un interés creciente en el uso de algoritmos de aprendizaje automático para mejorar la toma de decisiones médicas y predecir los resultados de los pacientes.

En la actualidad, hay muchas enfermedades que deben identificarse en sus fases iniciales para iniciar los tratamientos pertinentes. De lo contrario, podrían ser incurables y mortales. Por este motivo, es necesario analizar datos médicos complejos, informes médicos e imágenes médicas en menos tiempo pero con mayor precisión. Incluso hay casos en los que ciertas anomalías no pueden ser reconocidas directamente por los humanos.

En este tipo de situaciones, en las que es necesario realizar un análisis crucial de los datos médicos para revelar relaciones ocultas o anormalidades que no son visibles para los humanos, se utilizan enfoques de aprendizaje automático para la toma de decisiones informáticas en el ámbito sanitario. Implementar algoritmos para realizar estas tareas es difícil, pero lo que lo hace aún más complicado es aumentar la precisión del algoritmo al tiempo que se reduce el tiempo necesario para su ejecución.

Al principio, el procesamiento de grandes cantidades de datos médicos era una tarea importante que dio lugar a la adaptación del aprendizaje automático al ámbito biológico. Desde entonces, los campos de la biología y la biomedicina han alcanzado cotas más altas explorando más conocimientos e identificando relaciones que nunca antes se habían observado. La preocupación por tratar a los pacientes no sólo en función del tipo de enfermedad, sino también de su genética, está alcanzando su punto álgido, lo que se conoce como medicina de precisión. Cada día se realizan y prueban modificaciones en los algoritmos de aprendizaje automático para mejorar su rendimiento a la hora de analizar y presentar información más precisa.

En el ámbito sanitario, desde la extracción de información de documentos médicos hasta la predicción o el diagnóstico de una enfermedad, el aprendizaje automático ha estado implicado.

La imagen médica es un apartado que ha mejorado mucho con la integración de algoritmos de aprendizaje automático en el campo de la biología computacional. Hoy en día, muchos diagnósticos de enfermedades se realizan mediante el procesamiento de imágenes médicas utilizando algoritmos de aprendizaje automático. Además, la atención al paciente, la asignación de recursos y la investigación sobre tratamientos para diversas enfermedades también se están llevando a cabo utilizando la toma de decisiones computacional basada en el aprendizaje automático.

A lo largo de este artículo, se analizarán diversos algoritmos y enfoques de aprendizaje automático que se están utilizando para la toma de decisiones en el sector sanitario, junto con la implicación del aprendizaje automático en las aplicaciones sanitarias en el contexto actual. Con el conocimiento explorado, se hizo evidente que los métodos de aprendizaje profundo basados en redes neuronales han funcionado extremadamente bien en el campo de la biología computacional con el apoyo de la alta potencia de procesamiento de los sofisticados ordenadores modernos y se están aplicando ampliamente debido a su alta precisión de predicción y fiabilidad. Si se tiene en cuenta el panorama general combinando las observaciones, se observa que la biología computacional y la toma de decisiones biomédicas en el ámbito de la atención sanitaria han pasado a depender de algoritmos de aprendizaje automático, por lo que no pueden separarse del campo de la inteligencia artificial.

Predicción y detección de enfermedades

Se han implementado varias herramientas de aprendizaje automático para predecir o detectar una enfermedad en sus primeras fases, de modo que su tratamiento sea menos complejo y aumente la probabilidad de curación del paciente. Como resultado de estos enfoques, se han detectado diferentes tipos de enfermedades, pero con distintos niveles de precisión en función de factores como el algoritmo utilizado, el conjunto de características, el conjunto de datos de entrenamiento, etcétera.

1. **Cáncer:** El cuerpo humano tiene un número determinado de células de cada tipo. El cáncer comienza con cambios bruscos en la organización celular. Las señales que generan las células determinan el control y la división celular. Cuando estas señales fallan, las células se multiplican demasiado y forman un bulto llamado tumor. Hoy en día, la termografía (*técnica que permite determinar temperaturas a distancia y sin necesidad de contacto físico con el objeto a estudiar*) es más fiable, ya que no es invasiva ni ionizante. Con esta tecnología se ha ido produciendo resultados eficaces y positivos que la han hecho superior a otras tecnologías. A partir de las imágenes termográficas, con el uso de técnicas de extracción de características y técnicas de aprendizaje automático, se puede detectar la presencia de células cancerígenas. En este apartado se puede incluir distintos tipos como: **cáncer de mama, cáncer de pulmón, leucemia.**

2. **Diabetes:** La diabetes es una enfermedad crónica, y debe identificarse en las fases iniciales para su correcta medicación. La diabetes se produce cuando aumenta la proporción de azúcar en la sangre. Esto complica la vida de los pacientes por muchas razones. La diabetes puede clasificarse en tres tipos: diabetes 1, diabetes 2 y diabetes de gestación.

   Uno de los métodos utilizados son las redes neuronales. En este método, se entrenó una red neuronal feed-forward mediante un algoritmo de retropropagación. En este enfoque, las características que se tuvieron en cuenta son el número de embarazos, el grosor de los pliegues cutáneos, la insulina sérica, el IMC, el DPF y la edad, y la principal característica considerada fue el nivel de glucosa plasmática. Se observó que las predicciones realizadas mediante el uso de redes neuronales mostraron una mayor precisión, en comparación con

otros algoritmos de aprendizaje automático. También se ha investigado el uso de redes neuronales profundas (DNN) para la predicción de la diabetes mediante el entrenamiento de las DNN con validación cruzada quíntuple y decádruple. Cabe destacar que los dos enfoques antes mencionados que se adoptaron utilizando redes neuronales han mostrado una precisión cercana al 97 % en la predicción de la diabetes.

3. **Enfermedades cardiovasculares:** Las cardiopatías son enfermedades graves causadas por la obstrucción de las arterias del corazón. La cardiopatía crónica es el aumento de la placa en el interior de las arterias coronarias. Progresa lentamente y puede provocar un infarto. El metabolismo peculiar de la glucosa, la presión arterial extrema, la dislipidemia, el tabaquismo, la falta de ejercicio físico y la edad son algunos de los factores de riesgo identificados para las cardiopatías graves. Los síntomas de una cardiopatía pueden incluir dificultad para respirar, debilidad física, pies hinchados y fatiga con signos relacionados, etc.

   Muchas enfermedades cardiovasculares tienen sus raíces en la genética. Por lo tanto, las soluciones que utilizan la medicina de precisión se consideran más productivas especialmente en este tipo de enfermedades. La CNN, la red neuronal recurrente (RNN), el procesamiento del lenguaje natural (NLP), la SVM y la memoria a corto plazo (LSTM) son algunas de las técnicas de aprendizaje automático que podrían utilizarse de forma eficiente para crear CDSS (*sistemas de apoyo a la toma de decisiones clínicas*) precisos mediante el aprendizaje profundo.

4. **Enfermedad renal crónica:** La ERC es un tipo de enfermedad renal que afecta gradualmente a la funcionalidad del riñón y conduce a la insuficiencia renal. Puede diagnosticarse mediante datos clínicos, pruebas de laboratorio, estudios de imagen y biopsia. Pero ésta tiene algunas desventajas, como ser invasiva, costosa, lenta y, a veces, arriesgada. Aquí es donde puede aplicarse el aprendizaje automático para superar las desventajas mencionadas. En muchas predicciones de enfermedades mediante aprendizaje automático, SVM era un clasificador comúnmente utilizado. Pero en el caso de la ERC, no se han encontrado muchos estudios que utilicen SVM para la clasificación. RNA, DT y LR fueron los principales clasificadores de aprendizaje automático utilizados en este ámbito. Al observar los resultados obtenidos, la RNA mostró un rendimiento mucho mejor que el DT y el LR en el diagnóstico de la ERC.

5. **Parkinson:** El Parkinson es un trastorno del movimiento crónico y progresivo. No tiene causas, ni cura permanente, y las opciones de tratamiento son limitadas. Se ha descubierto que la enfermedad se debe a una producción reducida de dopamina, una sustancia química que controla el movimiento y la coordinación. Temblores, rigidez, lentitud de movimientos e inestabilidad postural son algunos de los síntomas. El movimiento anormal de contorsión es uno de los síntomas más graves de esta enfermedad. Algunos investigadores han aplicado algoritmos de aprendizaje automático en grabaciones de vídeo y visión por ordenador para diferenciar diagnóstico en las pruebas médicas de los pacientes con Parkinson. También se han utilizado muestras de voz. El Parkinson pertenece a la categoría de enfermedades neurodegenerativas que pueden afectar directa o indirectamente a las células cerebrales, lo que a su vez afecta al movimiento, al habla y a otras partes cognitivas.

6. **Enfermedades dermatológicas:** Las enfermedades de este tipo son complejas y presentan una gran variedad, y las personas tienen escasos conocimientos sobre ellas. Siempre es preferible la detección precoz, ya que podría tener consecuencias graves. El eccema, el herpes, el melanoma y la psoriasis son algunas de las enfermedades dermatológicas que deben identificarse en una fase temprana para poner la vida fuera de peligro.

   En uno de los enfoques adoptados para diagnosticar enfermedades dermatológicas, la primera fase consistió en la recopilación de datos y su ampliación mediante imágenes. La

segunda fase es muy importante, ya que en ella se crea y entrena el modelo. En la última fase, la imagen se convierte en una matriz y las características se descomponen utilizando el modelo entrenado que se ha creado.

Como muestra la figura 2.1, el número de artículos que aplican el *machine learning* al ámbito médico ha aumentado exponencialmente, sobre todo en lo que respecta al diagnóstico y el descubrimiento de fármacos. Según datos de Accenture, las aplicaciones vitales de inteligencia artificial en el ámbito médico-sanitario pueden generar un ahorro anual de 150.000 millones de dólares para el sector sanitario estadounidense en 2026.

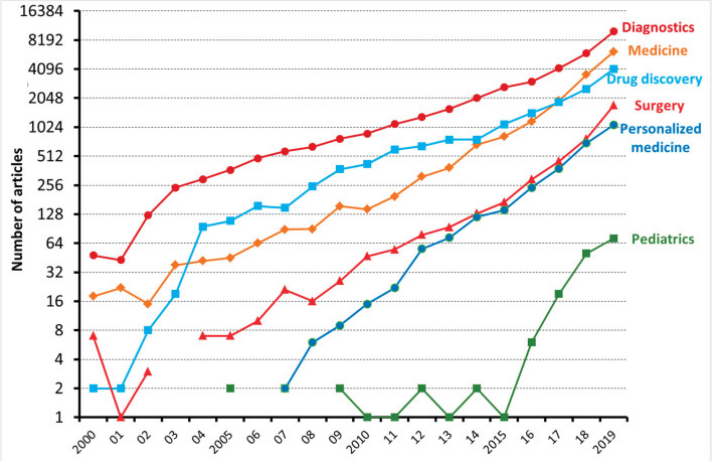

Figura 2.1: Evolución árticulos publicados

3. OBJETIVOS

En este estudio realizaremos una búsqueda exhaustiva por encontrar un modelo que nos consiga predecir lo mas preciso posible si un paciente ingresado por COVID-19 desarrollará o no eventos trombóticos.

La edad, el sexo y las enfermedades cardiovasculares se han relacionado con complicaciones tromboembólicas y peores resultados en COVID-19. Nuestra hipótesis es que, con ayuda de distintos algoritmos de *deep learning*, podemos ayudar a realizar un diágnostico preciso, aún desconocido, sobre las consecuencias de la enfermedad.

El enfoque principal será hallar una relación entre los pacientes ingresados por COVID-19 y los eventos trombóticos que desarrollan.

Partiremos de un *paper* publicado por unos cardiólogos del Hospital La Paz. No consiguieron demostrar ninguna correlación entre los trombos y el COVID-19. Trataremos de mejorar el modelo y comparar resultados y conclusiones que consigamos con nuestro estudio. Usaremos una base de datos común, hecha a partir de pacientes que tuvieron que ser hospitalizados a causa del virus.

También aprovecharemos nuestro estudio para investigar a fondo sobre otras enfermedades producidas por el virus (neumonías, cardiopatías...), distintos mecanismos de inteligencia artificial que se estén usando actualmente en el campo de la medicina, ayudar al estado del arte y personalmente, profundizar y aprender sobre estos dos conceptos tan amplios como son las redes neuronales y los eventos trombóticos en un cuerpo humano. Como último objetivo, pero no menos importante, trataremos de fomentar el uso de esta tecnología en la medicina ayudando a los médicos y pacientes.

4. METODOLOGÍA

Para poder lograr el objetivo del presente trabajo seguiremos un claro orden. En primer lugar nos informaremos del estado del parte del proyecto. Buscaremos y leeremos todos los trabajos y *papers* relacionados con la medicina y el machine learning.

En segundo lugar, profundizaremos nuestras nociones en la medicina y más concreto en eventos trombóticos en personas con COVID-19.

Se realizó un curso sobre análisis de datos durante el grado. El conocimiento respecto a las redes neuronales se ampliará mediante cursos online.

En última instancia probaremos muchos modelos en Python para conseguir nuestro objetivo principal.

4.1. Herramientas

En esta sección llevaremos a cabo una explicación sobre las distintas herramientas que vamos a emplear en nuestro estudio.

Para comenzar explicaremos los distintos lenguajes de programación empleados. Los empleados, a parte de su sencillez, han sido escogidos porque partimos de una base aprendida durane la carrera. **RStudio** es un entorno de desarrollo integrado para R, un lenguaje de programación para cálculo estadístico y gráficos. Lo usaremos para leer nuestra base de datos original y transformarla a un formato más legible. Con RStudio es fácil trabajar bases estadísticas como el *.dta* del que partimos.

Python es un lenguaje de programación interpretado, orientado a objetos, de alto nivel y semántica dinámica. Sus estructuras de datos incorporadas de alto nivel, combinadas con la tipificación dinámica y la vinculación dinámica, lo hacen muy atractivo para el desarrollo rápido de aplicaciones, así como para su uso como lenguaje de *scripting o glue* para conectar componentes existentes entre sí. La sintaxis de Python, sencilla y fácil de aprender, favorece la legibilidad y, por tanto, reduce el coste de mantenimiento de los programas. Python admite módulos y paquetes, lo que fomenta la modularidad del programa y la reutilización del código. Por ser un lenaguaje fácil de usar y muy popular para las aplicaciones que lo usaremos es la mejor elecció para el estudio.

Un editor de código es una herramienta que se utiliza para escribir y editar código. Suelen ser ligeros y pueden ser estupendos para aprender. Sin embargo, una vez que tu programa se hace más grande, necesitas probar y depurar tu código, ahí es donde entran los IDEs.

Un IDE (Entorno de Desarrollo Integrado) entiende tu código mucho mejor que un editor de texto. Suele ofrecer funciones como la automatización de la compilación, el análisis de código, las pruebas y la depuración. Esto puede acelerar considerablemente tu trabajo.

Visual Studio Code (VS Code) es un IDE gratuito y de código abierto creado por Microsoft que usaremos para el desarrollo de Python. VS Code es ligero y está repleto de potentes funciones. Esta es la razón por la que se está haciendo popular entre los desarrolladores de Python.

Una biblioteca Python es una colección de módulos relacionados. Contiene paquetes de código que pueden utilizarse repetidamente en diferentes programas. Hace que la programación en

Python sea más simple y conveniente para el programador. Ya que no necesitamos escribir el mismo código una y otra vez para diferentes programas. Las bibliotecas de Python juegan un papel muy importante en los campos de aprendizaje automático, ciencia de datos, visualización de datos, etc. Las librerías principales que usaremos son las siguientes:

- Numpy
- Pandas
- TensorFlow
- Keras
- Matplotlib

Figura 4.1: Librerías Python

5. BASE TEÓRICA

5.1. COVID-19

Historia

Los coronavirus son un grupo diverso de virus que infectan a muchos animales diferentes, y pueden causar infecciones respiratorias de leves a graves en los seres humanos. En 2002 y 2012, respectivamente, dos coronavirus altamente patógenos de origen zoonótico, el coronavirus del síndrome respiratorio agudo severo (SARS-CoV) y el coronavirus del síndrome respiratorio de Oriente Medio (MERS-CoV), surgieron en humanos y causaron enfermedades respiratorias mortales, lo que convirtió a los coronavirus emergentes en una preocupación de salud pública. A finales de 2019, un nuevo coronavirus designado como SARS-CoV-2 surgió en la ciudad de Wuhan, China, y causó un brote de neumonía viral inusual. Al ser altamente transmisible, esta nueva enfermedad por coronavirus, también conocida como enfermedad por coronavirus 2019 (COVID-19), se ha extendido rápidamente por todo el mundo.

A finales de diciembre de 2019, varios centros sanitarios de Wuhan, en la provincia china de Hubei, reportaron varios grupos de pacientes con neumonía de causa desconocida. De manera similar a los pacientes con SARS y MERS, estos pacientes mostraron síntomas de neumonía viral, incluyendo fiebre, tos y malestar en el pecho, y en casos graves disnea e infiltración pulmonar bilateral. Entre los primeros 27 pacientes hospitalizados, la mayoría de los casos se relacionaron con el mercado de marisco de Huanan, situado en el centro de Wuhan, que no sólo vende marisco, sino también animales vivos, incluidas aves de corral y animales salvajes. Según un estudio retrospectivo, el inicio del primer caso conocido se remonta al 8 de diciembre de 2019. El 31 de diciembre, la Comisión Municipal de Salud de Wuhan anunció un brote de neumonía de causa no identificada e informó a la Organización Mundial de la Salud (OMS).

El brote de COVID-19 en China alcanzó su pico en febrero 2020. Según la Comisión Nacional de Salud de China, el número total de casos siguió aumentando bruscamente a principios de febrero a un ritmo medio de más de 3.000 nuevos casos confirmados al día. Para controlar el COVID-19, China puso en marcha medidas estrictas. El 23 de enero se cerró la ciudad de Wuhan y se bloquearon todos los viajes y transportes que conectaban con la ciudad. En las dos semanas siguientes, se restringieron todas las actividades y reuniones al aire libre, y se cerraron las instalaciones en la mayoría de las ciudades. Gracias a estas medidas, el número diario de nuevos casos en China empezó a disminuir de forma constante.

Sin embargo, a pesar de la disminución de casos en China, la expansión internacional de COVID-19 se aceleró a partir de finales de febrero. La alta eficiencia de transmisión del SARS-CoV-2 y la abundancia de viajes internacionales permitieron la rápida propagación mundial de COVID-19. El 11 de marzo de 2020, la OMS calificó oficialmente el brote mundial de COVID-19 como una pandemia. Desde marzo, mientras que el COVID-19 en China estaba efectivamente controlado, el número de casos en Europa, Estados Unidos y otras regiones aumentó considerablemente. Según el Centro de Ciencia e Ingeniería de Sistemas de la Universidad Johns Hopkins, hasta el 11 de agosto de 2020, 216 países y regiones de los seis continentes habían notificado más de 20 millones de casos de COVID-19, y más de 733.000 pacientes habían fallecido. La elevada mortalidad se produjo especialmente cuando los recursos sanitarios se vieron desbordados. Estados Unidos es el país con el mayor número de casos hasta el momento.

Aunque las pruebas genéticas sugieren que el SARS-CoV-2 es un virus natural que proba-

blemente se originó en los animales, todavía no hay ninguna conclusión sobre cuándo y dónde entró el virus por primera vez en los seres humanos. Un estudio realizado en Francia detectó el SARS-CoV-2 por PCR en una muestra almacenada de un paciente que tuvo neumonía a finales de 2019, lo que sugiere que el SARS-CoV-2 podría haberse propagado allí. Sin embargo, ésto no puede dar una respuesta sólida al origen del SARS-CoV-2, por lo que no se puede excluir un resultado falso positivo. Para abordar esta cuestión tan controvertida, es necesario realizar más investigaciones retrospectivas que incluyan un mayor número de muestras de pacientes, animales y entornos en todo el mundo con ensayos bien validados.

Actualidad

La mayoría de las personas infectadas por el virus experimentarán una enfermedad respiratoria de leve a moderada y se recuperarán sin requerir un tratamiento especial. Sin embargo, algunas enfermarán gravemente y requerirán atención médica. Las personas mayores y las que padecen enfermedades subyacentes, como las cardiovasculares, la diabetes, las enfermedades respiratorias crónicas o el cáncer, tienen más probabilidades de desarrollar una enfermedad grave.

La actual pandemia de COVID-19 supone un importante reto para la mayoría de los sistemas sanitarios de todo el mundo. Mientras que la insuficiencia respiratoria sigue siendo el motivo más común de ingreso en las unidades de cuidados críticos y de muerte, la infección por SARS-CoV-2 ha demostrado ser una condición compleja con afectación multiórganica.

Los datos desde el comienzo sugieren una prevalencia significativa de factores de riesgo cardiovasculares entre los pacientes hospitalizados y en estado crítico con COVID-19. La edad y la enfermedad cardiovascular subyacente se asocian con peores resultados y las complicaciones tromboembólicas desempeñan un papel clave en el curso clínico de estos pacientes.

Trombos y COVID-19

Durante la pandemia ocasionada por el SARS-CoV2, se ha observado un aumento de enfermedades trombóticas tanto arteriales como venosas que parece estar en relación con diversos factores:

- La **inmovilidad** durante el periodo de confinamiento favorece el flujo sanguíneo más lento a nivel de las piernas, algo que en personas predispuestas, puede condicionar la aparición de trombos en las venas.
- El **miedo a acudir al hospital** en ocasiones ha contribuido al retraso en el diagnóstico de estas enfermedades que de detectarse a tiempo pueden prevenirse o tratarse en fases más tempranas y menos graves.
- La propia **infección por el virus** genera un estado de inflamación en el organismo de las personas afectadas por COVID19 que llega a favorecer la aparición de alteraciones en el sistema circulatorio y de trombos, por lo que en este tipo de pacientes, sería conveniente, realizar una valoración específica que permita detectar dichas complicaciones.

Los pacientes con infección por COVID-19 suelen tener un mayor riesgo de eventos trombóticos. La elección de antitrombóticos y las dosis se están estudiando actualmente en ensayos y estudios. Existe la necesidad de una estratificación individualizada del riesgo de eventos trombóticos para ayudar a los médicos en la toma de decisiones sobre la anticoagulación.

Tipos de trombo

Un trombo es un coágulo de sangre que se produce en el interior de un vaso sanguíneo y que dificulta o impide la circulación. Pueden producirse trombos tanto en las arterias, que llevan la circulación desde el corazón hasta los diferentes órganos, como en las venas, que conducen la sangre de retorno desde los órganos hacia el corazón.

Ocasionan distintas complicaciones según el nivel al que afecten:

- **Trombosis venosa:** Sin duda la complicación más grave de la patología trombótica venosa es el desplazamiento de los trombos a otros territorios como los pulmones, donde su repercusión puede ocasionar fallo respiratorio, cardiaco e incluso la muerte si no se detectan y tratan a tiempo.
- **Trombosis arterial:** En cuanto a los trombos en territorio arterial, su oclusión condiciona falta de riego con el desarrollo de infartos, más frecuentes a nivel cardiaco y cerebral, cuya gravedad y posibles repercusiones son bien conocidas.

Estas oclusiones pueden dar lugar a distintas enfermedades según el nivel al que afecten como ictus, infartos, isquemias, trombosis venosa profunda o tromboembolismo pulmonar.

Se ha tratado de identificar los factores de riesgo de estos eventos en los pacientes con COVID-19, lo que podría ayudar a los médicos en la prevención, la identificación temprana y el manejo de la enfermedad en los pacientes con COVID-19 hospitalizados y mejorar los resultados clínicos en estos pacientes.

5.2. Inteligencia Artificial

La inteligencia artificial (IA) se refiere a la simulación de la inteligencia humana en máquinas programadas para pensar como humanos e imitar sus acciones. El término también puede aplicarse a cualquier máquina que presente rasgos asociados a una mente humana, como el aprendizaje y la resolución de problemas.

La característica ideal de la inteligencia artificial es su capacidad para racionalizar y emprender acciones que tengan la mejor oportunidad de lograr un objetivo específico.

Un subconjunto de la inteligencia artificial es el aprendizaje automático *(Machine Learning)*, que se refiere al concepto de que los programas informáticos pueden aprender automáticamente de los nuevos datos y adaptarse a ellos sin la ayuda de los humanos.

Las técnicas de aprendizaje profundo *(Deep Learning)* permiten este aprendizaje automático mediante la absorción de enormes cantidades de datos no estructurados, como texto, imágenes o vídeo.

5.2.1. Machine Learning

El *Machine Learning* (aprendizaje automático) es una disciplina dentro de la Inteligencia Artificial que, mediante algoritmos, proporciona a los ordenadores la capacidad de identificar

patrones a partir de datos masivos para realizar predicciones. Este método de aprendizaje permite a los ordenadores realizar tareas específicas de forma autónoma, es decir, sin necesidad de ser programados.

El término se utilizó por primera vez en 1959. Sin embargo, ha cobrado relevancia en los últimos años debido al aumento de los recursos informáticos y al enorme incremento de la cantidad de datos. Las técnicas de aprendizaje automático son, de hecho, una parte fundamental del *Big Data*.

Existen distintos **tipos de algoritmos** en *Machine Learning*:

- ***Supervised learning:*** estos algoritmos llevan incorporado un aprendizaje previo y se basan en un sistema de etiquetas asociado a los datos que les permite tomar decisiones o hacer predicciones. Un ejemplo es un detector de spam que etiqueta un correo electrónico como spam o no, en función de los patrones que ha aprendido del historial de correos electrónicos (remitente, relación texto/imagen, palabras clave del asunto, etc.).
- ***Unsupervised learning:*** estos algoritmos no tienen conocimientos previos. Se enfrentan a un caos de datos con el objetivo de encontrar patrones que permitan de alguna manera su organización. Por ejemplo, en el campo del marketing se utilizan para extraer patrones de los datos masivos obtenidos de las redes sociales y crear campañas publicitarias muy segmentadas.
- ***Reinforced learning:*** su objetivo es que un algoritmo aprenda de su propia experiencia. Es decir, que sea capaz de tomar la mejor decisión en diferentes situaciones según un proceso de prueba y error en el que se premian las decisiones correctas. Actualmente se utiliza para permitir el reconocimiento facial, realizar diagnósticos médicos o clasificar secuencias de ADN.

5.2.2. Deep Learning

El *Deep Learning*(aprendizaje profundo) es una técnica de aprendizaje automático que enseña a los ordenadores a hacer lo que es natural para los humanos: aprender a base del ejemplo. El aprendizaje profundo es la tecnología clave detrás de los coches autónomos, que les permite reconocer una señal de STOP o distinguir a un peatón de una farola. Es la clave del control por voz en dispositivos como móviles, tabletas, televisores y altavoces. El aprendizaje profundo está recibiendo mucha atención últimamente y con razón. Está consiguiendo resultados que antes no eran posibles.

En el aprendizaje profundo, un modelo informático aprende a realizar tareas de clasificación directamente a partir de imágenes, texto o sonido. Los modelos de aprendizaje profundo pueden lograr una precisión de vanguardia, que a veces supera el rendimiento de los seres humanos. Los modelos se entrenan utilizando un gran conjunto de datos etiquetados y arquitecturas de redes neuronales que contienen muchas capas.

Algoritmos Machine Learning and Deep Learning

El aprendizaje profundo es un subconjunto del aprendizaje automático, y este último es un subconjunto de la inteligencia artificial. Se puede pensar en ellos como una serie de círculos concéntricos superpuestos, en los que la IA ocupa el mayor, seguida del aprendizaje automático

y, a continuación, el aprendizaje profundo. En otras palabras, el aprendizaje profundo es IA, pero la IA no es aprendizaje profundo.

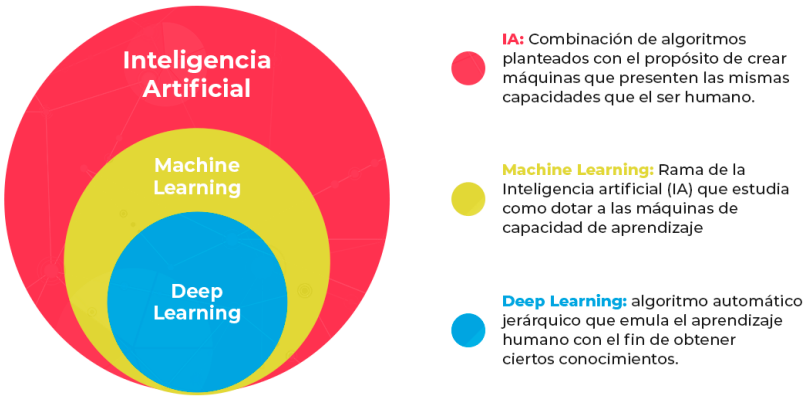

Figura 5.1: Subconjuntos IA

En resumen, el aprendizaje automático es la IA que puede adaptarse automáticamente con una actuación humana mínima. El aprendizaje profundo es un subconjunto del aprendizaje automático que utiliza redes neuronales artificiales para imitar el proceso de aprendizaje del cerebro humano.

Machine Learning	**Deep Learning**
Un subconjunto de la IA	Un subconjunto del Machine Learning
Puede entrenarse con conjuntos de datos más pequeños	Requiere grandes cantidades de datos
Requiere más intervención humana para corregir y aprender	Aprende por sí mismo del entorno y de errores pasados
Entrenamiento más corto y menor precisión	Entrenamiento más largo y mayor precisión
Establece correlaciones simples y lineales	Establece correlaciones complejas y no lineales

Tabla 5.1: Diferencias entre Machine Learning y Deep Learning

En la actualidad existen muchos tipos diferentes de algoritmos de Machine Learning, algunos de los cuales pueden ayudar a los ordenadores a jugar al ajedrez, realizar operaciones quirúrgicas y ser más inteligentes y personales.

En los últimos cinco años, los *Data Scientists* han construido sofisticadas máquinas de tratamiento de datos mediante la ejecución de técnicas avanzadas. Los resultados han sido asombrosos.

Los algoritmos aprendizaje automático están automatizados y se automodifican para seguir mejorando con el tiempo. Existen 3 tipos distintos vistos anteriormente: *Supervised, Unsupervised Learning y Reinforcement Learning.* Destacamos los más populares:

- *Linear Regression*
- *Logistic Regression*
- *Decision Tree*
- *SVM (Support Vector Machine) Algorithm*
- *Naive Bayes Algorithm*
- *KNN (K- Nearest Neighbors) Algorithm*

- *K-Means*
- *Random Forest Algorithm*
- *Dimensionality Reduction Algorithms*
- *Gradient Boosting Algorithm and AdaBoosting Algorithm*

Por otro lado, el aprendizaje profundo ha ganado una popularidad enorme en la computación científica, y sus algoritmos son ampliamente utilizados por industrias que resuelven problemas complejos. Todos los algoritmos de aprendizaje profundo utilizan distintos tipos de redes neuronales para realizar tareas específicas.

Los algoritmos de aprendizaje profundo entrenan máquinas aprendiendo de ejemplos. Sectores como la sanidad, *eCommerce*, el entretenimiento y la publicidad utilizan habitualmente el aprendizaje profundo.

Aunque los algoritmos de aprendizaje profundo presentan representaciones de autoaprendizaje, dependen de las RNA(Redes Neuronales Artificiales) que reflejan la forma en que el cerebro computa la información. Durante el proceso de entrenamiento, los algoritmos utilizan elementos desconocidos en la distribución de entrada para extraer características, agrupar objetos y descubrir patrones de datos útiles. De forma muy similar al entrenamiento de máquinas para el autoaprendizaje, esto ocurre en múltiples niveles, utilizando los algoritmos para construir los modelos.

Los modelos de aprendizaje profundo hacen uso de varios algoritmos. Aunque ninguna red se considera perfecta, algunos algoritmos son más adecuados para realizar tareas específicas. Para elegir los adecuados, es bueno adquirir una comprensión de todos los algoritmos primarios:

- *Convolutional Neural Networks (CNNs)*
- *Long Short Term Memory Networks (LSTMs)*
- *Recurrent Neural Networks (RNNs)*
- *Generative Adversarial Networks (GANs)*
- *Radial Basis Function Networks (RBFNs)*
- *Multilayer Perceptrons (MLPs)*
- *Self Organizing Maps (SOMs)*
- *Deep Belief Networks (DBNs)*
- *Restricted Boltzmann Machines (RBMs)*
- *Autoencoders*

5.2.3. Redes neuronales

De todos los algoritmos vistos, nosotros nos profundizaremos en un algortimo del *Deep Learning*, las Redes Neuronales.

Inspiración biológica

El cerebro está compuesto principalmente por unas 10.000 millones de neuronas, cada una conectada a otras 10.000 neuronas. Cada una de las manchas amarillas de la figura 5.2 son cuerpos celulares neuronales (soma), y las líneas son los canales de entrada y salida (dendritas y axones) que las conectan.

Cada neurona recibe entradas electroquímicas de otras neuronas en las dendritas. Si la suma de estas entradas eléctricas es lo suficientemente potente como para activar la neurona, ésta transmite una señal electroquímica a lo largo del axón y pasa esta señal a las otras neuronas cuyas dendritas están unidas en cualquiera de los terminales del axón. Estas neuronas pueden entonces dispararse.

Es importante señalar que una neurona se dispara sólo si la señal total recibida en el cuerpo celular supera un determinado nivel. La neurona se dispara o no se dispara, pero no hay diferentes grados de disparo.

Así pues, todo nuestro cerebro está compuesto por estas neuronas transmisoras electroquímicas interconectadas. A partir de un gran número de unidades de procesamiento extremadamente simples (cada una de las cuales realiza una suma ponderada de sus entradas y dispara una señal binaria si la entrada total supera un determinado nivel), el cerebro consigue realizar tareas extremadamente complejas.

Este es el modelo en el que se basan las redes neuronales artificiales. Hasta ahora, las redes neuronales artificiales ni siquiera se han acercado a modelar la complejidad del cerebro, pero han demostrado ser buenas en problemas que son fáciles para un ser humano pero difíciles para un ordenador tradicional, como el reconocimiento de imágenes y las predicciones basadas en conocimientos pasados.

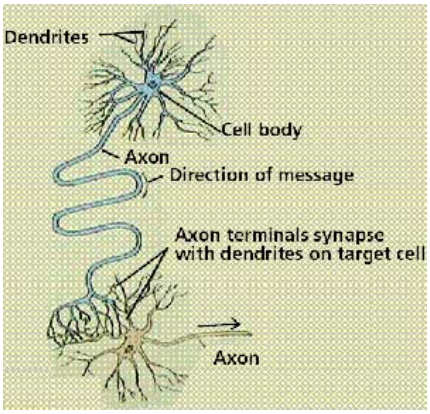

Figura 5.2: Neurona biológica

El Perceptrón

El perceptrón es un modelo matemático de una neurona biológica. Mientras que en las neuronas reales la dendrita recibe señales eléctricas de los axones de otras neuronas, en el perceptrón estas señales eléctricas se representan como valores numéricos. En las sinapsis entre la dendrita y los axones, las señales eléctricas se modulan en distintas cantidades. Esto también se modela en el perceptrón multiplicando cada valor de entrada por un valor denominado peso. Una neurona real dispara una señal de salida sólo cuando la intensidad total de las señales de

entrada supera un determinado umbral. En un perceptrón, este fenómeno se modela calculando la suma ponderada de las entradas para representar la intensidad total de las señales de entrada y aplicando una función escalonada a la suma para determinar la salida. Como en las redes neuronales biológicas, esta salida se transmite a otros perceptrones.

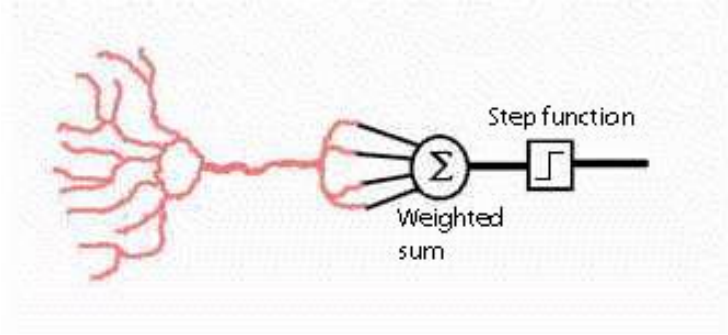

Figura 5.3: Perceptrón

Para describir las redes neuronales se utiliza habitualmente una serie de términos. Se enumeran en la tabla siguiente:

El vector de entrada	Todos los valores de entrada de cada perceptrón se denominan colectivamente vector de entrada.
El vector de pesos	Todos los valores de peso de cada perceptrón se denominan colectivamente el vector de peso.

Como ya se ha mencionado, un perceptrón calcula la suma ponderada de los valores de entrada. Para simplificar, supongamos que hay dos valores de entrada, x e y para un determinado perceptrón P. Dejemos que los pesos para x e y sean A y B respectivamente, la suma ponderada podría representarse como: $A \cdot x + B \cdot y$.

Dado que el perceptrón emite un valor distinto de cero sólo cuando la suma ponderada supera un cierto umbral C, se puede escribir la salida de este perceptrón de la siguiente manera:

$$Output\ P = \begin{cases} 1 & A \cdot x + B \cdot y > C \\ 0 & A \cdot x + B \cdot y \leqslant C \end{cases}$$

Recordemos que $A \cdot x + B \cdot y > C$ y $A \cdot x + B \cdot y \leqslant$ son las dos regiones del plano XY separadas por la recta $A \cdot x + B \cdot y = C$. Si consideramos la entrada (x, y) como un punto de un plano, el perceptrón nos dice a qué región del plano pertenece ese punto. Estas regiones, al estar separadas por una única línea, se denominan regiones linealmente separables.

Este resultado es útil porque resulta que algunas funciones lógicas, como los operadores booleanos AND, OR y NOT, son linealmente separables, es decir, que pueden realizarse con un único perceptrón. Podemos ilustrar (para el caso 2D) por qué son linealmente separables representando cada una de ellas en un gráfico:

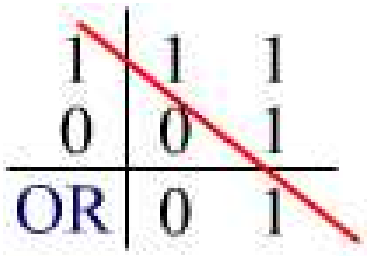

Figura 5.4: Funciones lógicas linealmente separables

En los gráficos anteriores, los dos ejes son las entradas que pueden tomar el valor 0 o 1, y los números del gráfico son la salida esperada para una entrada concreta. Utilizando un vector de pesos adecuado para cada caso, un único perceptrón puede realizar todas estas funciones.

Sin embargo, no todos los operadores lógicos son linealmente separables. Por ejemplo, el operador XOR no es linealmente separable y no puede ser realizado por un único perceptrón. Sin embargo, este problema podría superarse utilizando más de un perceptrón dispuestos en redes *feed-forward*.

Historia: 40s a la actualidad

Las redes neuronales han tenido una historia única en el ámbito de la tecnología. A diferencia de muchas tecnologías actuales, que fracasan de inmediato o se popularizan de inmediato, las redes neuronales fueron populares durante un breve periodo de tiempo, hicieron una pausa de dos décadas y, desde entonces, no han dejado de ser populares.

En 1943, el neurofisiólogo Warren McCulloch y el matemático Walter Pitts escribieron un artículo sobre el posible funcionamiento de las neuronas. Para describir cómo podrían funcionar las neuronas del cerebro, modelaron una red neuronal sencilla utilizando circuitos eléctricos. En 1949, Donald Hebb escribió *The Organization of Behavior* (La organización del comportamiento), una obra que señalaba el hecho de que las vías neuronales se **fortalecen cada vez que se utilizan**, un concepto fundamentalmente esencial para las formas en que aprenden los seres humanos. Si dos nervios se activan al mismo tiempo, la conexión entre ellos se refuerza.

En los años 50, con el desarrollo de los ordenadores, por fin fue posible simular una hipotética red neuronal. El primer paso en este sentido lo dio Nathanial Rochester, de los laboratorios de investigación de IBM. El primer intento fracasó.

En 1959, Bernard Widrow y Marcian Hoff, de Stanford, desarrollaron modelos llamados *ADALINE* y *MADALINE*. En un alarde típico del amor de Stanford por los acrónimos, los nombres proceden de su uso de *Multiple ADAptive LINear Elements* (elementos de enlace adaptativos múltiples). *ADALINE* se desarrolló para reconocer patrones binarios, de modo que si leía bits de una línea telefónica, podía predecir el siguiente bit. *MADALINE* fue la primera red neuronal aplicada a un problema del mundo real, utilizando un filtro adaptativo que elimina los ecos en las líneas telefónicas. Aunque el sistema es igual de antiguo como los sistemas de control del tráfico aéreo, aunque al igual que éstos, sigue utilizándose comercialmente.

En 1962, Widrow y Hoff desarrollaron un procedimiento de aprendizaje que examina el valor anterior al peso y lo ajusta (es decir, 0 ó 1) según la regla $Cambio\,de\,peso = (Valor\,de\,la\,linea\,anterior\,al\,peso) \cdot (Error/(Número\,de\,entradas))$. Se basa en la idea de que mientras un perceptrón activo puede tener un gran error, se pueden ajustar los valores de peso

para distribuirlo por toda la red, o al menos a los perceptrones adyacentes. La aplicación de esta regla sigue dando lugar a un error si la línea anterior al peso es 0, aunque con el tiempo se corregirá. Si el error se conserva de forma que se distribuya a todos los pesos, el error se elimina.

A pesar del éxito posterior de la red neuronal, la arquitectura tradicional de von Neumann se apoderó de la escena informática y la investigación neuronal quedó relegada. Irónicamente, el propio John von Neumann sugirió imitar las funciones neuronales utilizando relés telegráficos o tubos de vacío.

En el mismo período de tiempo, se escribió un artículo que sugería que no podía haber una extensión de la red neuronal de una sola capa a una red neuronal de múltiples capas. Además, muchas personas en este campo estaban utilizando una función de aprendizaje que era fundamentalmente defectuosa porque no era diferenciable en toda la línea. Como consecuencia, la investigación y la financiación disminuyeron drásticamente.

A esto se unió el hecho de que los primeros éxitos de algunas redes neuronales llevaron a exagerar el potencial de las redes neuronales, sobre todo teniendo en cuenta la tecnología práctica de la época. Las promesas no se cumplieron y, en ocasiones, las grandes cuestiones filosóficas provocaron temor. Los escritores reflexionaron sobre el efecto que las llamadas "máquinas pensantes"tendrían sobre los humanos, ideas que siguen vigentes hoy en día.

La idea de un ordenador que se programe a sí mismo es muy atractiva. Si el Windows 2000 de Microsoft pudiera reprogramarse a sí mismo, podría reparar los miles de errores que cometió el personal de programación. Estas ideas eran atractivas, pero muy difíciles de llevar a la práctica. Además, la arquitectura von Neumann estaba ganando popularidad. Se produjeron algunos avances en este campo, pero en su mayor parte las investigaciones fueron escasas.

En 1972, Kohonen y Anderson desarrollaron una red similar de forma independiente, de la que hablaremos más adelante. Ambos utilizaron matemáticas matriciales para describir sus ideas, pero no se dieron cuenta de que lo que estaban haciendo era crear una matriz de circuitos analógicos *ADALINE*. Se supone que las neuronas activan un conjunto de salidas en lugar de una sola.

En 1975 se desarrolló la primera red multicapa, una red no supervisada. Luego, en 1982 se renovó el interés por este campo. John Hopfield, de Caltech, presentó un trabajo ante la Academia Nacional de Ciencias. Su planteamiento consistía en crear máquinas más útiles utilizando líneas bidireccionales. Antes, las conexiones entre neuronas eran unidireccionales. En el mismo año, Reilly y Cooper utilizaron una *red híbrida* con múltiples capas, cada una de las cuales utilizaba una estrategia de resolución de problemas diferente.

También en 1982, se celebró una conferencia conjunta EE.UU.-Japón sobre Redes Neuronales Cooperativas/Competitivas. Japón anunció un nuevo descubrimiento de Quinta Generación en redes neuronales, y las ponencias de EE.UU. generaron preocupación por la posibilidad de que EE.UU. quedara rezagado en este campo. (La informática de quinta generación implica inteligencia artificial. La primera generación utilizaba interruptores y cables, la segunda el transistor, la tercera tecnología de estado sólido, como circuitos integrados y lenguajes de programación de alto nivel, y la cuarta generadores de código). Como consecuencia, hubo más financiación y, por tanto, más investigación en este campo.

En 1986, con las redes neuronales de múltiples capas de actualidad, el problema era cómo extender la regla de Widrow-Hoff a múltiples capas. Tres grupos independientes de investigadores, uno de los cuales incluía a David Rumelhart, antiguo miembro del departamento de psicología de Stanford, propusieron ideas similares que ahora se denominan redes de retropropagación porque

distribuye los errores de reconocimiento de patrones por toda la red. Las redes híbridas utilizaban sólo dos capas, estas redes de retropropagación utilizan muchas. El resultado es que las redes de retropropagación son *aprendices lentos*, que necesitan posiblemente miles de iteraciones para aprender.

En la actualidad, las redes neuronales se utilizan en diversas aplicaciones. La idea fundamental que subyace de las redes neuronales es que si funciona en la naturaleza, debe poder funcionar en los ordenadores, sin embargo, el futuro de las redes neuronales está en el desarrollo del hardware. Al igual que las máquinas avanzadas que juegan al ajedrez, como *Deep Blue*, las redes neuronales rápidas y eficientes dependen de que el hardware esté especificado para su uso final.

La investigación centrada en el desarrollo de redes neuronales es relativamente lenta. Debido a las limitaciones de los procesadores, las redes neuronales tardan semanas en aprender. Algunas empresas están intentando crear lo que se denomina un *compilador de silicio* para generar un tipo específico de circuito integrado optimizado para la aplicación de redes neuronales. Los distintos tipos de chips que se están desarrollando son digitales, analógicos y ópticos. Uno podría descartar inmediatamente las señales analógicas como cosa del pasado. Sin embargo, las neuronas del cerebro funcionan más como señales analógicas que como señales digitales. Mientras que las señales digitales tienen dos estados distintos (1 o 0, encendido o apagado), las señales analógicas varían entre valores mínimos y máximos. Sin embargo, puede que pase un tiempo antes de que los chips ópticos puedan utilizarse en aplicaciones comerciales.

Funcionamiento Redes Neuronales

Como hemos estudiado previamente, las redes neuronales artificiales se inspiran en las neuronas biológicas del cuerpo humano, que se activan en determinadas circunstancias y, como respuesta, el cuerpo realiza una acción relacionada. Las redes neuronales artificiales están formadas por varias capas de neuronas artificiales interconectadas, alimentadas por funciones de activación que ayudan a activarlas y desactivarlas. Al igual que los algoritmos tradicionales, las redes neuronales aprenden determinados valores en la fase de entrenamiento.

En pocas palabras, cada neurona recibe una versión multiplicada de entradas y pesos aleatorios, a la que se añade un valor de sesgo estático (único para cada capa neuronal); a continuación, se pasa a una función de activación adecuada que decide el valor final que debe emitir la neurona. Existen varias funciones de activación según la naturaleza de los valores de entrada. Una vez generada la salida de la última capa de la red neuronal, se calcula la función de pérdida (entrada frente a salida) y se realiza la retropropagación, en la que los pesos se ajustan para que la pérdida sea mínima. La operación general se centra en encontrar los valores óptimos de los pesos.

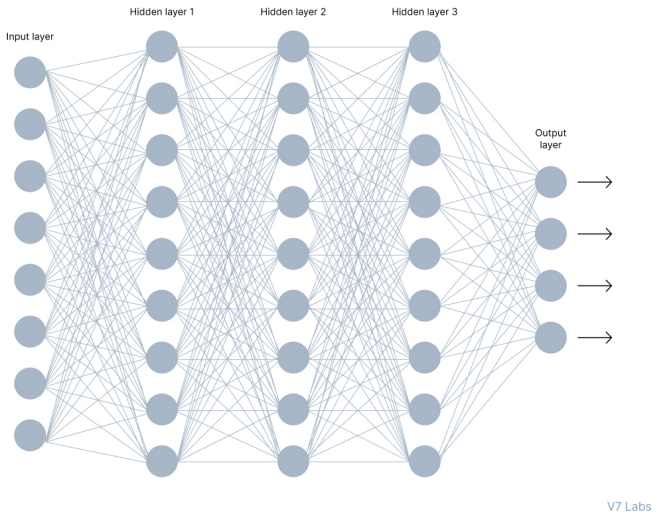

Figura 5.5: Arquitectura de redes neuronales

En la imagen de arriba puedes ver una red neuronal formada por neuronas interconectadas. Cada una de ellas se caracteriza por su peso, sesgo y función de activación. La **capa de entrada** recibe datos brutos del dominio. En esta capa no se realiza ningún cálculo. Los nodos sólo transmiten la información (características) a la **capa oculta**. Ésta realiza todo tipo de cálculos sobre las características introducidas a través de la capa de entrada y transfiere el resultado a la **capa de salida**, que entrega el valor final como resultado.

Cada nodo individual es su propio modelo de regresión lineal. Una vez determinada una capa de entrada, se asignan **pesos**. Estas ponderaciones ayudan a determinar la importancia de cualquier variable, ya que las más grandes contribuyen de forma más significativa a la salida en comparación con otras entradas. A continuación, todas las entradas se multiplican por sus respectivos pesos y se suman. Después, la salida pasa por una función de activación, que determina la salida.

$$\sum w_i \cdot x_i + bias = w_1 \cdot x_1 + w_2 \cdot x_2 + w_3 \cdot x_3 + bias$$

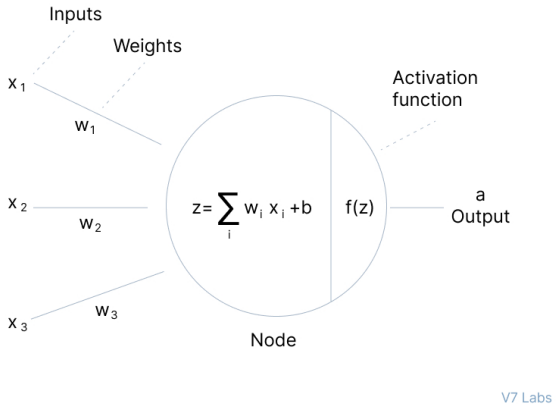

Figura 5.6: Funcionamiento un nodo

La **función de activación** decide si una neurona debe activarse o no. Esto significa que decidirá si la entrada de la neurona a la red es importante o no en el proceso de predicción utilizando operaciones matemáticas más sencillas. El papel principal de la función de activación es transformar la entrada ponderada sumada del nodo en un valor de salida para ser alimentado a la siguiente capa oculta o como salida.

El propósito de una función de activación es añadir no linealidad a la red neuronal.

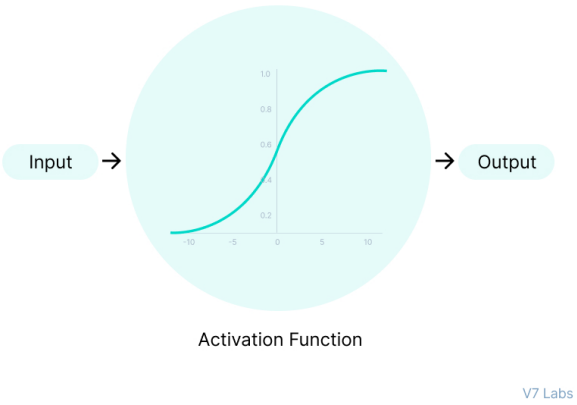

Figura 5.7: Función de activación

Supongamos que tenemos una red neuronal que funciona sin las funciones de activación.

En ese caso, cada neurona sólo realizará una transformación lineal en las entradas utilizando los pesos y los sesgos. Esto se debe a que no importa cuántas capas ocultas añadamos en la red neuronal; todas las capas se comportarán de la misma manera porque la composición de dos funciones lineales es una función lineal en sí misma.

Aunque la red neuronal se simplifique, el aprendizaje de cualquier tarea compleja es imposible, y nuestro modelo no sería más que un modelo de regresión lineal. Vamos a ver las más importantes:

1. **Función escalón:** La entrada alimentada a la función de activación se compara con un determinado umbral; si la entrada es mayor que éste, la neurona se activa; en caso contrario, se desactiva, lo que significa que su salida no pasa a la siguiente capa oculta.

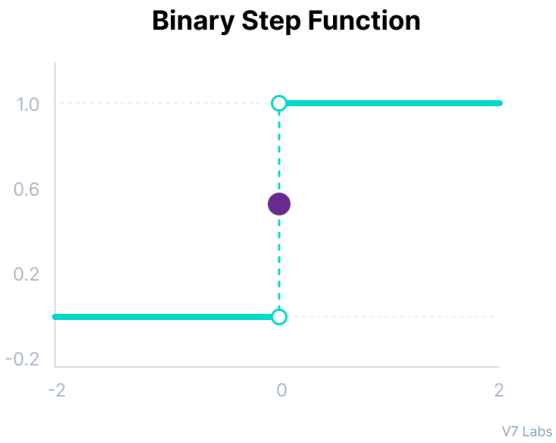

Figura 5.8: Función escalón

$$f(x) = \begin{cases} 1 & for\ x > 0 \\ 0 & for\ x \leqslant 0 \end{cases}$$

2. **Función sigmoide:** Esta función toma cualquier valor real como entrada y emite valores en el rango de 0 a 1.

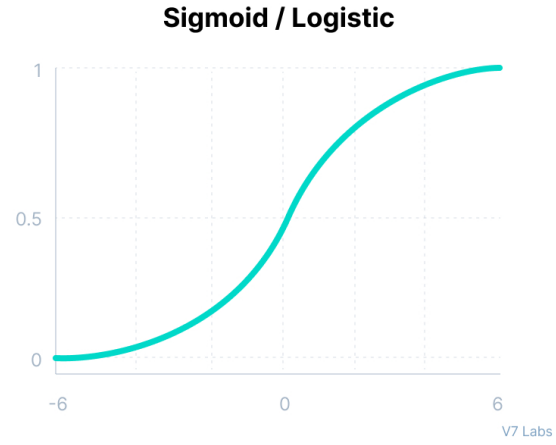

Figura 5.9: Función sigmoide

$$f(x) = \frac{1}{1+e^{-x}}$$

3. **Función Tanh (tangente hiperbólica):** La función Tanh es muy similar a la función de activación sigmoide, e incluso tiene la misma forma de S con la diferencia en el rango de salida de -1 a 1.

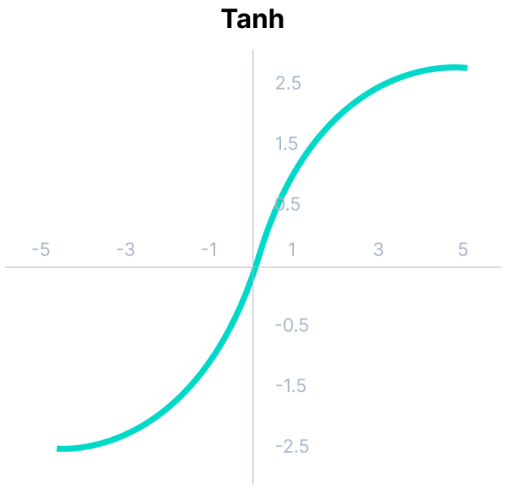

Figura 5.10: Función Tanh

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

4. **Función ReLU:** ReLU son las siglas de *Rectified Linear Unit* (unidad lineal rectificada). Aunque da la impresión de ser una función lineal, ReLU tiene una función derivada y permite la retropropagación, al tiempo que la hace eficiente desde el punto de vista computacional. El principal diferenciador es que la función ReLU no activa todas las neuronas al mismo tiempo. Las neuronas sólo se desactivarán si la salida de la transformación lineal es menor que 0.

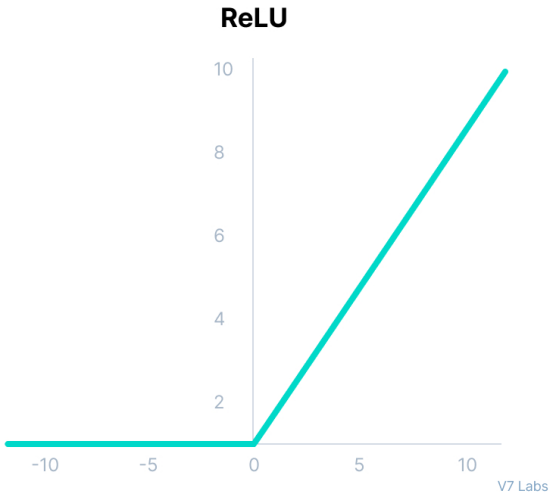

Figura 5.11: Función ReLu

$$f(x) = max(0, x)$$

Se debe ajustar la función de activación de la capa de salida en función del tipo de problema de predicción que estás resolviendo, es decir, el tipo de variable predicha.

Entrenamiento: Feedforward vs. Backpropagation

Feedforward y *backpropagation* son dos conceptos fundamentales en el entrenamiento de redes neuronales.

Feedforward se refiere al proceso de pasar los datos de entrada a través de la red, capa por capa, para generar una salida prevista. Este proceso comienza en la capa de entrada, donde los datos de entrada son transformados por los pesos y sesgos de la capa para producir un conjunto de activaciones. Estas activaciones pasan a la siguiente capa, donde se transforman de nuevo, y así sucesivamente hasta que las activaciones llegan a la capa de salida, donde se utilizan para generar la predicción final.

Backpropagation, por su parte, es una técnica utilizada para calcular el gradiente de la función de pérdida con respecto a los pesos y sesgos de la red. El algoritmo de optimización utiliza este gradiente para ajustar los pesos y los sesgos con el fin de minimizar la pérdida y mejorar las predicciones de la red. Backpropagation comienza en la capa de salida, donde se calcula el error entre la salida prevista y la salida real. Este error se propaga hacia atrás a través de la red, capa por capa, utilizando la regla de la cadena para calcular el gradiente de la función de pérdida con respecto a los pesos y sesgos en cada capa.

Entrenar redes neuronales para que funcionen bien en una tarea determinada requiere un ajuste cuidadoso de la arquitectura y los hiperparámetros de la red, así como un conocimiento profundo de conceptos clave del entrenamiento como las funciones de pérdida y el descenso de gradiente. En este apartado veremos una visión general de estos conceptos y discutiremos su papel en el entrenamiento de redes neuronales.

Al entrenar el modelo, querremos evaluar su precisión utilizando una función de coste (*loss function*). Existen varias funciones. Una se conoce como error cuadrático medio (*MSE = Mean Squared Error*).

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y}_i)^2$$

Otra que veremos es error medio absoluto (*MAE = Mean Absolute Error*).

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \widehat{y}_i|$$

En última instancia, el objetivo es minimizar nuestra función de coste para garantizar la corrección del ajuste para cualquier observación dada. A medida que el modelo ajusta sus ponderaciones y sesgos, utiliza la función de coste y el aprendizaje por refuerzo para alcanzar el punto de convergencia, o el mínimo local. El proceso por el que el algoritmo ajusta sus ponderaciones es a través del descenso gradiente.

Éste es un algoritmo de optimización que se utiliza para minimizar la función de pérdidas moviéndose iterativamente en la dirección de descenso más pronunciado definida por el negativo del gradiente.

$$\mathbf{a}_{n+1} = \mathbf{a}_n - \gamma \nabla f(\mathbf{a}_n)$$

En el aprendizaje automático, utilizamos el descenso gradiente para actualizar los parámetros de nuestro modelo. Los parámetros son los pesos en las redes neuronales.

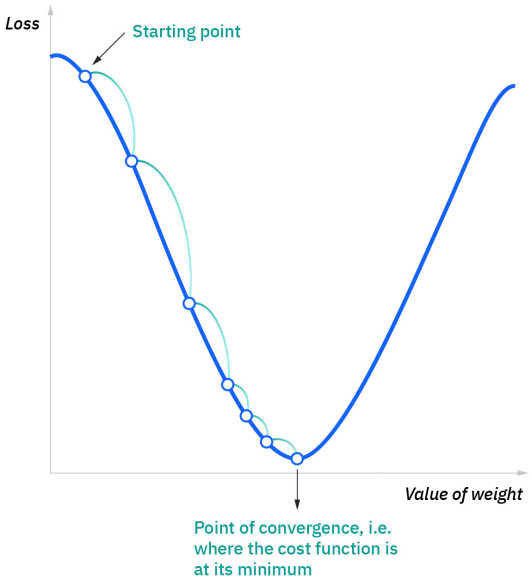

Figura 5.12: Minimizar LOSS

El tamaño de los pasos que da el descenso gradiente en dirección al mínimo local viene determinado por la tasa de aprendizaje, que determina lo rápido o lento que nos moveremos hacia los pesos óptimos.

Para que el algoritmo de descenso por gradiente alcance el mínimo local, debemos fijar la tasa de aprendizaje en un valor adecuado, que no sea ni demasiado bajo ni demasiado alto. Esto es importante porque si los pasos que da son demasiado grandes, puede que no alcance el mínimo local porque rebota una y otra vez entre la función convexa de descenso gradiente (izquierda figura 5.15). Si fijamos la tasa de aprendizaje en un valor muy pequeño, el descenso gradiente acabará alcanzando el mínimo local, pero puede tardar un poco (derecha figura 5.15).

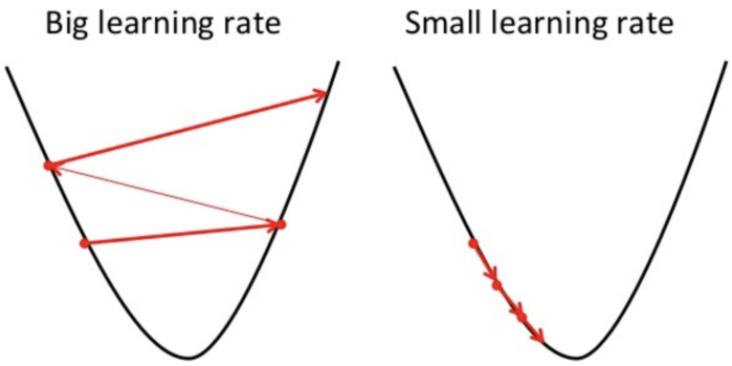

Figura 5.13: Tamaño de los pasos

El descenso de gradiente estocástico (**SGD**) trata de ser una aproximación estocástica de la optimización por descenso de gradiente, ya que sustituye el gradiente real (calculado a partir de todo el conjunto de datos) por una estimación del mismo (calculada a partir de un subconjunto

de datos seleccionados aleatoriamente). Especialmente en problemas de optimización de altas dimensiones, esto reduce la elevada carga computacional, consiguiendo iteraciones más rápidas a cambio de una menor tasa de convergencia.

5.3. Aspectos relevantes del código

Ya se han explicado en el marco teórico las redes neuronales y algunos de los posibles parámetros para su entrenamiento, en la práctica existen algunos más que discutiremos en este apartado. Muchos de estos parámetros se consiguen optimizar a prueba y error, pero dependiendo del objetivo de un problema y del tipo, se puede acortar esa búsqueda.

Función de coste: MSE vs MAE

MAE es la media de todos los errores absolutos. Por otro lado, MSE mide la media de los cuadrados de los errores. Se tratan de funciones de pérdidas que dan resltados distintos. Ahora bien, ¿cuándo usar uno u otro?

Predicción	**Actual**	**Error absoluto**	**Error cuadrático**
140	120	20	400
80	50	30	900
10	10.2	0.2	0.04
1	0.95	0.05	0.0025

Tabla 5.2: Diferencia de errores

Debido al cuadrado, los errores grandes se acentúan y tienen un efecto relativamente mayor en el valor de la métrica de rendimiento. Al mismo tiempo, el efecto de los errores relativamente pequeños será aún menor. A veces, esta propiedad del error al cuadrado se denomina penalizar los errores extremos o ser susceptible a los valores atípicos. Según la aplicación, esta propiedad puede considerarse positiva o negativa. Por ejemplo, enfatizar los errores grandes puede ser una medida discriminatoria deseable en la evaluación de modelos.

Tanto MAE como MSE son buenas métricas generales, pero cada una tiene sus puntos fuertes y débiles. En última instancia, cuál es mejor depende del objetivo del proyecto. Si desea entrenar un modelo centrado en reducir los errores atípicos, la mejor opción es el MSE, mientras que si esto no es importante y prefiere una mayor interpretabilidad, entonces es mejor el MAE. A lo largo del apartado probaremos distintas combinaciones.

Cabe destacar otra función de perdidas: *Cross-Entroy*. El objetivo de la entropía cruzada es tomar las probabilidades de salida y medir la distancia respecto a los valores de verdad (como se muestra en la figura siguiente).

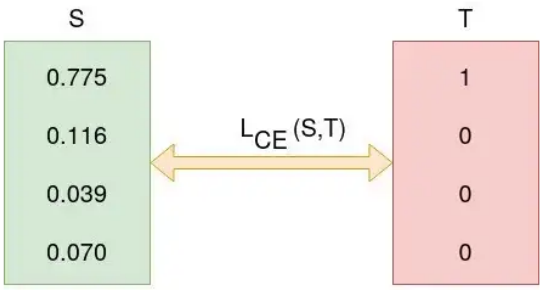

Figura 5.14: Cross Entropy

La entropía cruzada **binaria** compara cada una de las probabilidades previstas con el resultado real de la clase, que puede ser 0 ó 1. A continuación, calcula la puntuación que penaliza las probabilidades en función de la distancia al valor esperado. Es decir, lo cerca o lejos que están del valor real. Es una función incluida dentro de cross-entropy pero más precisa para nuestro caso.

Función de activación: Sigmoide, ReLu y Tanh

Ya visto previamente la función de activación decide si una neurona debe activarse o no. Esto significa que decidirá si la entrada de la neurona a la red es importante o no en el proceso de predicción utilizando operaciones matemáticas más sencillas.

El papel de la función de activación es derivar la salida de un conjunto de valores de entrada alimentados a un nodo (o una capa).

La elección de la función de activación en la capa oculta determinará la capacidad del modelo de red para aprender el conjunto de datos de entrenamiento. La elección de la función de activación en la capa de salida definirá el tipo de predicciones que puede hacer el modelo.

Cuando la función de activación de una neurona es una **sigmoidea** es una garantía de que la salida de esta unidad estará siempre entre 0 y 1. Ya que nuestro problema trata de clasificar en dos grupos los pacientes, los que sí desarrollaron eventos trombóticos(1) y los que no(0), la **capa de salida** siempre contendrá esta función.

Para las capas ocultas no queda tan clara la decisión. Hay tres funciones muy populares a considerar en nuestro entrenamiento: **ReLu, sigmoide y Tanh**.

La función de activación lineal rectificada, o función de activación **ReLU**, es quizás la función más común utilizada para las capas ocultas.

Es común porque es sencilla de implementar y eficaz a la hora de superar las limitaciones de otras funciones de activación populares anteriormente, como Sigmoid y Tanh. En concreto, es menos susceptible a los gradientes de fuga que impiden entrenar modelos profundos, aunque puede sufrir otros problemas. Aunque se probarán las otras dos, esta será la más usada.

Optimizador Adam

La elección del algoritmo de optimización para el modelo de aprendizaje profundo puede significar la diferencia entre buenos resultados en minutos, horas y días.

Adam es un algoritmo de optimización que puede utilizarse en lugar del procedimiento clásico de descenso de gradiente estocástico para actualizar los pesos de la red de forma iterativa basándose en los datos de entrenamiento. Algo interesante, Adam es el nombre que recibió de sus autores, no un acrónimo.

Los autores describen Adam como una combinación de las ventajas de otras dos extensiones del descenso de gradiente estocástico. En concreto:

- Algoritmo de Gradiente Adaptativo (AdaGrad) que mantiene una tasa de aprendizaje por parámetro que mejora el rendimiento en problemas con gradientes dispersos (por ejemplo, problemas de lenguaje natural y visión por ordenador).
- Root Mean Square Propagation (RMSProp), que también mantiene tasas de aprendizaje por parámetro que se adaptan en función de la media de las magnitudes recientes de los gradientes para el peso (por ejemplo, la rapidez con la que cambia). Esto significa que el algoritmo funciona bien en problemas en línea y no estacionarios (por ejemplo, ruidosos).

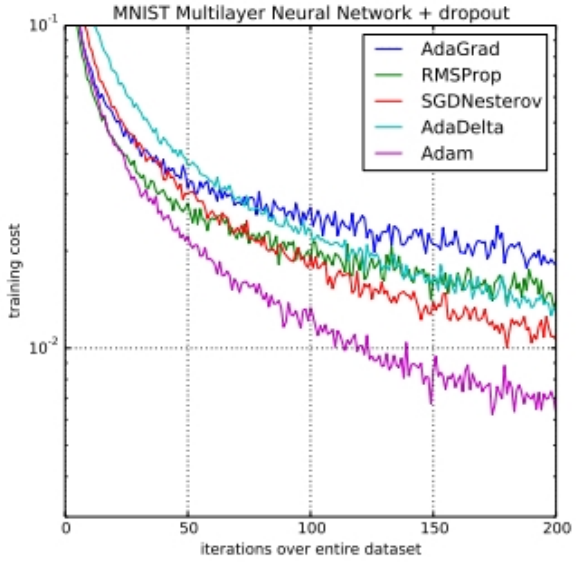

Figura 5.15: Comparación optimizadores

En un paper publicado en 2016 publicado por Sebastian Ruder Ruder, 2016, en la sección *Which optimizer to use?* recomendó usar Adam.

Insofar, RMSprop, Adadelta, and Adam are very similar algorithms that do well in similar circumstances. [...] its bias-correction helps Adam slightly outperform RMSprop towards the end of optimization as gradients become sparser. Insofar, Adam might be the best overall choice.

Número de neuronas y capas

El número de neuronas en la capa de entrada es igual al número de características de los datos, así que esta capa no tiene elección alguna. El número de neuronas en la salida será una por ser un problema de clasificador. Lo interesante llega en las capas ocultas, ¿es más mejor?.

Existen muchos métodos prácticos para determinar el número correcto de neuronas que se deben utilizar en las capas ocultas, como los siguientes:

- El número de neuronas ocultas debe estar entre el tamaño de la capa de entrada y el tamaño de la capa de salida.
- El número de neuronas ocultas debe ser 2/3 del tamaño de la capa de entrada, más el tamaño de la capa de salida.
- El número de neuronas ocultas debe ser inferior al doble del tamaño de la capa de entrada.

Además, el número de neuronas y de capas necesarias para la capa oculta también depende de los casos de entrenamiento, la cantidad de valores atípicos, la complejidad de los datos que deben aprenderse y el tipo de funciones de activación utilizadas. Por tanto iremos cambiando este parámetro y testando.

Criterio de parada

Uno de los principales retos a la hora de entrenar redes neuronales es cuánto tiempo entrenarlas.

Si el entrenamiento es insuficiente, el modelo no se ajustará bien a los conjuntos de datos de entrenamiento y de prueba. Si se entrena demasiado, el modelo se ajustará demasiado al conjunto de datos de entrenamiento y tendrá un rendimiento deficiente en el conjunto de pruebas.

Una solución intermedia es entrenar en el conjunto de datos de entrenamiento, pero detener el entrenamiento en el momento en que el rendimiento en un conjunto de datos de validación empiece a degradarse. Este método sencillo, eficaz y ampliamente utilizado para entrenar redes neuronales se denomina parada temprana.

Durante el entrenamiento, el modelo se evalúa en un conjunto de datos de validación después de cada época. Si el rendimiento del modelo en el conjunto de datos de validación empieza a degradarse (por ejemplo, la pérdida empieza a aumentar o la precisión empieza a disminuir), se detiene el proceso de entrenamiento.

Este procedimiento se denomina *early stopping* y es quizá una de las formas más antiguas y utilizadas de regularización de redes neuronales. Mediremos la métrica *val_loss* para determinar cuando parar de entrenar. Se han escogido un nivel de paciencia 10, que quiere decir que si tras 10 épocas no mejora esa métrica, para. Por tanto, emplearemos un alto número de *epochs* sabiendo que no tiene porque llegar a todas.

Datos estandarizados

Frecuentemente en los problemas de redes neuronales se escalan los datos. Esto se trata de obtener datos con media 0 y varianza 1. Buscamos que todos los datos tengan una magnitud similar.

Overfitting y Underfitting

Se dice que un modelo es un buen modelo de machine learning si clasifica adecuadamente cualquier dato de entrada nuevo del dominio del problema. Esto nos ayuda a hacer predicciones sobre datos futuros, que el modelo de datos nunca ha visto. Supongamos ahora que queremos comprobar lo bien que nuestro modelo aprende y predice los nuevos datos. Para ello, tenemos el sobreajuste(*overfitting*) y el infraajuste(*underfitting*), que son los principales responsables de los malos resultados de los algoritmos.

- **Underfitting:** Se dice que el modelo tiene underfitting cuando no puede capturar la tendencia subyacente de los datos, es decir, sólo funciona bien con los datos de entrenamiento, pero funciona mal con los datos de prueba. Éste destruye la precisión de nuestro modelo. Su aparición significa simplemente que nuestro modelo o el algoritmo no se ajusta lo suficientemente bien a los datos.

  Razones del underfitting:

  - Sesgo alto y varianza baja.
  - El tamaño del conjunto de datos de entrenamiento utilizado no es suficiente.
  - El modelo es demasiado simple.
  - Los datos de entrenamiento no están limpios y contienen ruido.

- **Overfitting:** Se dice que el modelo posee overfitting cuando no realiza predicciones precisas en los datos de prueba. Cuando un modelo se entrena con tantos datos, empieza a aprender del ruido y de las entradas de datos inexactos de nuestra base de datos y cuando lo pruebas con el conjunto de test dan como resultado una alta varianza. Entonces el modelo no categoriza los datos correctamente, debido al exceso de detalles y ruido.

  Razones del overfitting:

  - Alta varianza y bajo sesgo.
  - El modelo es demasiado complejo.
  - El tamaño de los datos de entrenamiento.

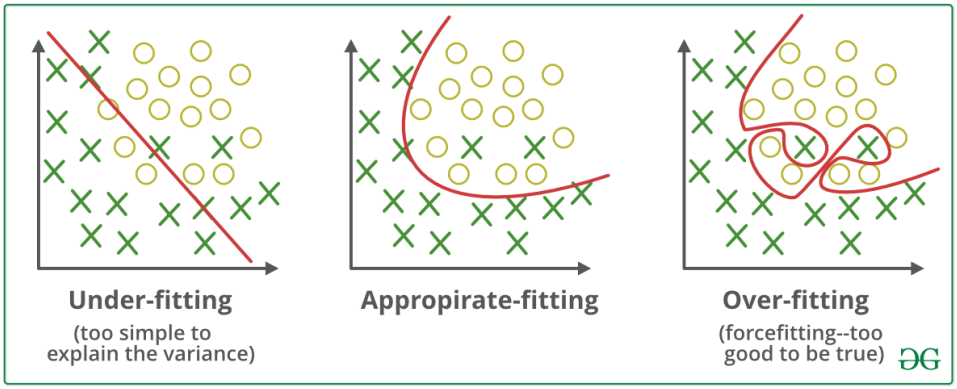

Figura 5.16: Visualización underfitting y overfitting

6. PREPARACIÓN DE DATOS

Se realizó un cribado de todos los pacientes consecutivos con sospecha clínica de COVID-19 atendidos en el Servicio de Urgencias de un centro de atención terciaria de Madrid desde el 1 de marzo de 2020 hasta el 20 de abril de 2020. Los pacientes sólo se incluyeron en el estudio si tenían confirmación de infección por SARS-CoV-2 mediante pruebas PCR. Se excluyeron del presente análisis los pacientes que estaban vivos y habían sido diagnosticados 30 días antes del cierre de la base de datos.

Se extrajeron datos epidemiológicos, demográficos, clínicos, de laboratorio, de tratamiento y de resultados de las historias clínicas electrónicas de los ingresos y posteriores ingresos hospitalarios utilizando un formulario electrónico estandarizado de recogida de datos. Además, se revisó el sistema central de historias clínicas, que recoge información e informes médicos de todos los hospitales públicos y centros de atención primaria de la Comunidad de Madrid, para obtener información adicional y realizar un seguimiento. Todos los datos fueron revisados exhaustivamente por un equipo de 13 cardiólogos. Se prestó especial atención a la identificación de las características basales y los resultados CV.

Variables	**Pacientes (n=3064)**	**Sin eventos trombóticos (n=2955)**	**Con eventos trombóticos (n=109)**
Edad	62.2	62.02	67.23
Sexo varón	1680 (54.83 %)	1610 (54.48 %)	70 (64.22 %)
Tabaco	301 (9.82 %)	290 (9,81 %)	11 (10.09 %)
Cáncer	301 (9.82 %)	288 (9.75 %)	13 (11.93 %)
Muerte	626 (20.54 %)	584 (19.76 %)	42 (38.54 %)

Tabla 6.1: Características de los pacientes del estudio con y sin episodios trombóticos

Podemos observar algunos números de interés respecto a nuestra base de datos. Algunos resultados que llaman la atención son: más hombres que mujeres desarrollaron un evento trombótico, en caso de desarrollar un evento trombótico, lógicamente la probabilidad de muerte es mayor y observamos que la edad es mayor en las personas que padecieron estas enfermedades.

Variables	**Pacientes (n=3064)**	**Supervivientes (n=2416)**	**No supervivientes (n=626)**
Edad	62.2	57.6	80.45
Sexo varón	1680 (54.83 %)	1274 (52.73 %)	396 (63.26 %)
Tabaco	301 (9.82 %)	219 (9,06 %)	79 (12.62 %)
Cáncer	301 (9.82 %)	189 (7.82 %)	111 (17.73 %)

Tabla 6.2: Características de los pacientes del estudio según su estado vital a 1 mes

En este análisis identificando por los pacientes que sobrevivieron a la enfermedad un mes después, llama mucho la atención la media de edad de los pacientes que murieron. Tener una edad alta y haber ingresado por COVID-19 vemos que tiene una gran mortalidad.

6.1. Limpieza base de datos

Nos encontramos ante una base de datos, que está lejos de ser óptima, para un modelo de redes neuronales. Disponemos de pocos datos para el entrenamiento de la red. Aún así servirá para entender y aprender sobre la relación de las redes neuronales y el COVID-19 que es el objetivo de este trabajo.

De los 3064 pacientes del estudio, muchos registros no están completos, es decir, con muchos valores *NaN*. A su vez, existen ciertas variables más incompletas que otras, que eliminaremos. La red neuronal, como ya se ha visto, aprende sola y asigna unos pesos a las variables importantes cuya influencia en la función de costes es mayor. Aún sabiendo esto último, se ha hablado con expertos en el ámbito de la medicina, y han sido eliminadas algunas de las variables manualmente.

Teniendo todo esto en cuenta y para depurar la base de datos para ser alimentada a nuestra red, nos quedamos con 2923 pacientes y 39 de 471 variables.

Edad	Historia de arritmias	Antecedente FA	Antecedente FTA
Antecedente MP/DAI	Hipertensión	Diabetes	Insuficiencia cardiaca
Cardiopatía isquémica	Prótesis valvular bio.	Prótesis valvular mec.	Cardiopatía congénita
Tabaco	EPOC	Asma	Obesidad
Cáncer	Hepatopatía	Insuficiencia renal	Anticoagulantes preingreso
Lopinavir	Corticoide	Evento vascular periférico	FA/FTA UVI
TV/FV UVI	Exitus	Sexo	Hemorragia grave
Bradicardia ingreso	TV/FV ingreso	Prótesis valvular	FA/FTA ingreso
Antecedente TV	Enfermedad vascular	Miocardiopatía	Inmunodepresión
Hospitalización	Bradicardia UVI	**TROMBO**	

Tabla 6.3: Variables pacientes

La tabla 6.3 muestra las variables seleccionadas de nuestro conjunto de datos. Algunas variables son numéricas como la edad, otras binarias. La mayoría tienen relación con si padecen o no una enfermedad o cualidad característica del paciente.

Algunos términos para la comprensión completa de los datos:

- **Arritmia:** Es un trastorno de la frecuencia cardíaca (pulso) o del ritmo cardíaco.
- **FA/FTA:** Fibrilación auricular, Flutter auricular.
- **MP/DAI:** Marcapasos. Desfibrilador automático implantable.
- **EPOC:** Enfermedad pulmonar obstructiva crónica.
- **Lopinavir:** Medicamento para tratar COVID-19.
- **TV/FV:** Taquicarida ventricular. Fibrilación ventricular.
- **Bradicardia:** Frecuencia cardíaca menor a 50 latidos por minuto.

Trombo será nuestra variable dependiente a predecir. Incluye los distintos eventos trombóticos ya estudiados.

7. RESULTADOS

7.1. Evaluación de modelos

El propósito de este sección es establecer una red neuronal artificial y, basándonos en los resultados obtenidos, ir modificando los parámetros con el objetivo de alcanzar una estructura óptima. Se ha hecho un estudio previo a estos modelos siguientes para determinar los parámetros correctos. En cada modelo se ha empleado distintas funciones de activación, funciones de coste, número de neuronas y capas.

7.1.1. Modelos con una capa oculta

Vamos a entrenar modelos con distintos parámetros pero todos con una capa oculta. Esto incluye, por supuesto, la capa de entrada y la capa oculta que siempre van a ser de las mismas dimensiones. Cambiaremos el distinto número de neuronas a lo largo de los distintos modelos.

7.1.1.1 Modelos MSE

Empezamos aprobar distintas estructuras de nuestra red neurona. El primer parámetro que modificaremos en este apartado será la función de coste *Mean Squared Error(MSE)*. Éste es común en los tres modelos siguientes.

Parámetros	
Nº Capas	3
F. Activación	ReLu + Sigmoid
F. Coste	MSE
Callbacks	Early Stopping
Optimizador	Adam

Tabla 7.1: Parámetros modelos MSE

El elemento diferenciador de cada modelo será el número de neuronas. Emplearemos las tres fórmulas vistas:

- $\frac{in+out}{2}$
- $\frac{2}{3} \cdot (in + out)$
- $< 2 \cdot in$

Modelo 1

El **primer modelo**, siguiendo a la primer fórmula vista, contará con una capa de entrada de 38 neuronas, una capa oculta con 20 neuronas y una capa de salida con una única salida. El objetivo de la última neurona será que devuelva un valor entre 0 y 1 que será la probabilidad de que el paciente desarrolle un evento trombótico.

Al echar un vistazo a la figura 7.1 se observa una clara divergencia de las lineas de test y validación a partir de la época 2. Es por esto que nuestro *callback* al ver que no mejoraba el error de validación en 10 épocas, decide poner un fin al entrenamiento de la red. Aún así vemos que el error de test seguía descendiendo. Éste es otro de los motivos por los que el *Early Stopping* es de gran utilidad, para evitar el *overfitting*.

Por tanto el error obtenido (MSE) en nuestro conjunto de validación al terminar el entrenamiento de la red es 0.0392. Con una precisión del 95.55 % somos capaces de predecir si un nuevo paciente, convaleciente por COVID-19, desarrollará un evento trombótico o no.

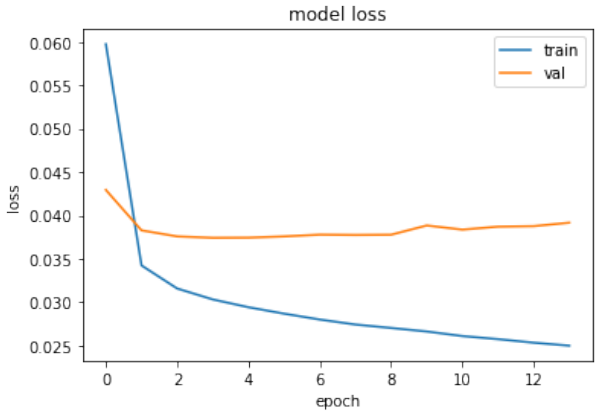

Figura 7.1: Modelo 1

Modelo 2

En el **segundo modelo** utilizamos una red neuronal con 26 neuronas en la capa oculta, un número ligeramente superior a la anterior red, obteniéndose así una estructura [38,26,1].

Observando la figura 7.2 se aprecia un cambio notable en el entrenamiento. Esta vez, las dos líneas de test y validación son coincidentes hasta la época 10, momento en el que comienzan a divergir. Viendo como nuestra línea de validación dejaba de mejorar, incluso ascendía, el ES decide poner un fin al entrenamiento en la época 19.

El error resultante en el conjunto de validación es de 0.0308. Cabe destacar que se está mencionando el error de la última época, si bien, habiendo conseguido errores menores en épocas previas. La precisión con la que predicimos el eventro trombótico de un paciente ingresado es del 96.69 %.

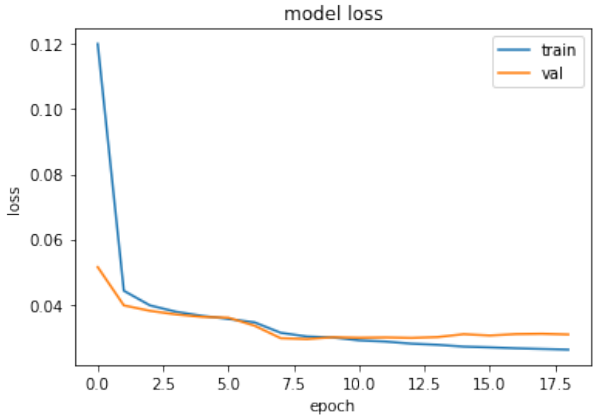

Figura 7.2: Modelo 2

Modelo 3

En el **último modelo** de nuestro uso de la función de costes MSE, probamos una red con 75 neuronas. La estructura de la que disponemos es [38,75,1]. Se han aumentado considerablemente las neuronas en este problema, discutimos la posible mejora en esta acción.

En la figura 7.3 obtenemos algo muy similar a 7.1. Dos líneas de entrenamiento que desde una época temprana divergen y no existe señal de que vayan a cortar. El entrenamiento del modelo fue parado en la época 20 para evitar un overfitting considerable en la red.

Conseguimos un error inferior al anterior modelo con menos neuronas, con un valor de 0.0354. La precisión esta vez resulta ser del 96.24 %, valor totalmente aceptable.

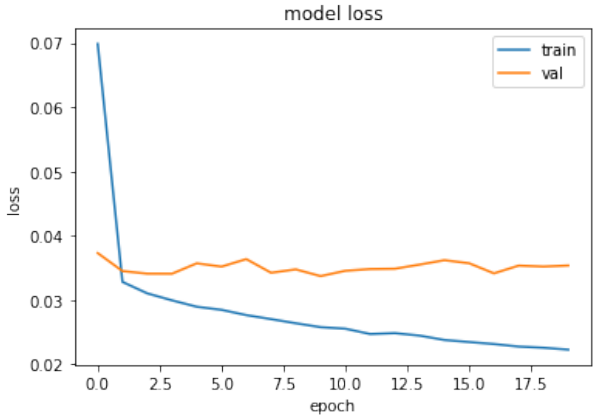

Figura 7.3: Modelo 3

La tabla 8.1 muestra una comparación de los tres modelos vistos. En todos se obtiene un error muy bajo con una alta precisión al predecir nuestro objetivo. Llama la atención, como el aumento de neuronas del segundo modelo, respecto del primero, se ha visto beneficiado en una disminución del error. Sin embargo, este aumento de neuronas ha provocado un aumento negativo en el error del tercer modelo, el cual tenía el mayor número de neuronas.

Los resultados no son congruentes para descartar la hipótesis de que un aumento considerable de las neuronas, no mejorará el modelo. A lo largo del estudio, mantendremos un enfoque en

esta hipótesis ya que de ser cierta o no, evitaría mucha complejidad y ahorro computacional.

Nº Neuronas	**Estructura**	**F. Coste**	**F. Activación**	**val_loss**	**Epochs**
$\frac{in+out}{2} = 20$	[38,20,1]	MSE	ReLu	0.0392	14
$\frac{2}{3} \cdot (in + out) = 26$	[38,26,1]	MSE	ReLu	0.0308	19
$< 2 \cdot in = 75$	[38,75,1]	MSE	ReLu	0.0354	20

Tabla 7.2: Tabla comparación modelos MSE

7.1.1.2 Modelos MAE

En esta sección probaremos todas las redes anteriores pero cambiando la función de coste con la que se entrena el modelo. *Mean Average Error* será la función. Igual que en el anterior apartado, los tres modelos que veremos siguen estos parámetros en común.

Parámetros	
Nº Capas	3
F. Activación	ReLu + Sigmoid
F. Coste	MAE
Callbacks	Early Stopping
Optimizador	Adam

Tabla 7.3: Parámetros modelos MAE

Diferenciaremos el número de neuronas en cada modelo y discutiremos el efecto de este cambio.

Modelo 4

En el **cuarto modelo** obtendremos así una estructura [38,26,1]. La gráfica 7.4 muestra un buen entrenamiento. Llama la atención como a partir de la época 5 actua como si existiese una asíntota en el valor 0.03. Aún así las dos líneas van a la par a lo largo del entrenamiento y esto es buena señal. En la época 27 se ha producido el *ES* aunque a juicio personal se podría haber detenido antes.

El error medio de la época final es del 0.0346 para el conjunto de validación. Con una precisión del 96.69 % conseguimos predecir el desarrollo de eventos trombóticos con este modelo.

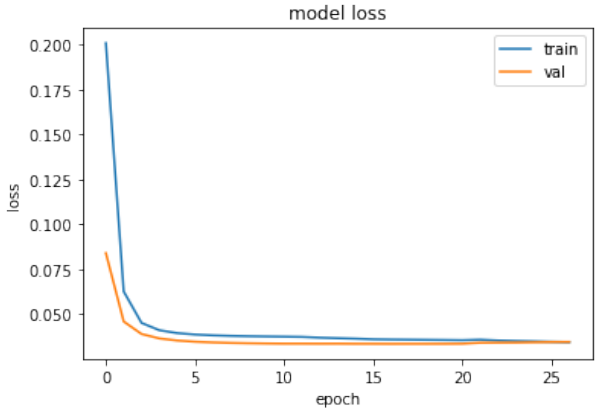

Figura 7.4: Modelo 4

Modelo 5

El **quinto modelo** dispondrá de una capa oculta de 26 neuronas. Un aumento en el número de neuronas del 30 % respecto del modelo anterior contemplado.

Una gráfica muy similar al modelo de 20 neuronas contempla en la figura 7.5. La parada ha ocurrido en la época 22. El error ha sido 0.0333 lo que implica una reducción respecto el error anterior por lo que el aumento de neuronas ha sido significativo en el modelo 5. Sin embargo la disminución ha sido tan baja, que la precisión sigue siendo del 96.69 %.

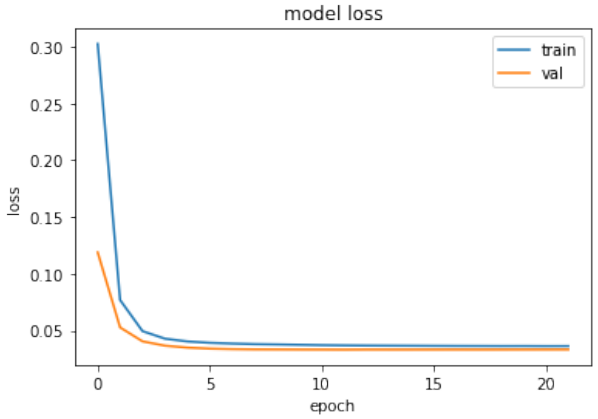

Figura 7.5: Modelo 5

Modelo 6

En el último **modelo 6** de la sección la red cuenta con 76 neuronas. La gráfica 7.6 no varía en cuanto a forma respecto a las otras. Esta vez el criterio de parada ha aguantado más tiempo de entrenamiento, hasta la época 49.

A diferencia de los modelos con medición MSE, el error en este sexto modelo es más bajo que los anteriores. Siendo el error 0.033 y la precisión manteniéndose en 96.69 %.

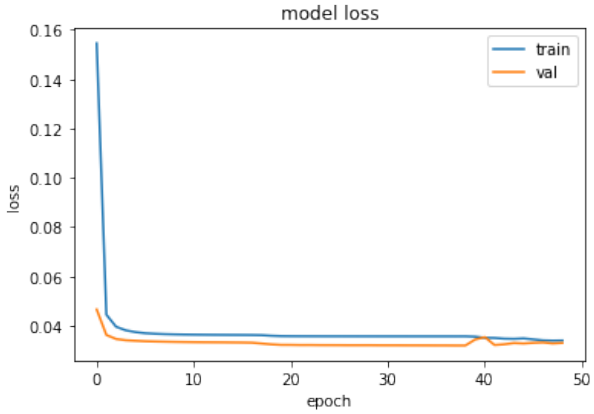

Figura 7.6: Modelo 6

En la tabla 7.4 inmediatamente inferior se recopilan los datos obtenidos de nuestros modelos evaluados con el error MAE.

Esta vez se ha conseguido una relación indirecta del número de capas y valor de error. Obtenemos en cada modelo un aumento de número de neuronas con respecto al anterior y una disminución del error. Sí se cumple la hipótesis presentada con anterioridad en los modelos MSE. Cabe recalcar que la precisión se ha mantenido contante en los tres modelos aún con una minúscula reducción en los errores.

Nº Neuronas	**Estructura**	**F. Coste**	**F. Activación**	**val_loss**	**Epochs**
$\frac{in+out}{2} = 20$	[38,20,1]	MAE	ReLu	0.0346	27
$\frac{2}{3} \cdot (in + out) = 26$	[38,26,1]	MAE	ReLu	0.0333	22
$< 2 \cdot in = 75$	[38,75,1]	MAE	ReLu	0.0330	49

Tabla 7.4: Tabla comparación modelos MAE

7.1.1.3 Modelos Binary Cross-Entropy

Por último vamos a probar modelos con la función de coste *Binary Cross-Entropy*. Los tres modelos siguientes siguen la misma estructura:

Parámetros	
Nº Capas	3
F. Activación	ReLu + Sigmoid
F. Coste	Binary Cross-Entropy
Callbacks	Early Stopping
Optimizador	Adam

Tabla 7.5: Parámetros modelos Binary Cross-Entropy

Modelo 7

En **primera instancia**, el modelo compondrá una primera capa de entrada con todas las variables, una segunda capa oculta con la función de coste y una última con la función sigmoid.

La figura 7.7 muestra una red inservible con un gran overfitting. Después de 14 épocas de entrenamiento, el callback lo detuvo. Se ve claramente la divergencia entre los dos conjuntos de datos y como el conjunto de entrenamiento cada vez ajustaba los pesos de la red más a su favor, sobre entrenando la red y empeorándola para el conjunto de validación.

Respecto al error, con esta función de coste obtenemos un error cinco veces mayor que los errores anteriores, 0.1611. Con una precisión del 95.6 %. Aunque el error sea notablemente superior, este cambio no se refleja en la precisión, si bien se mantiene del mismo orden de magnitud.

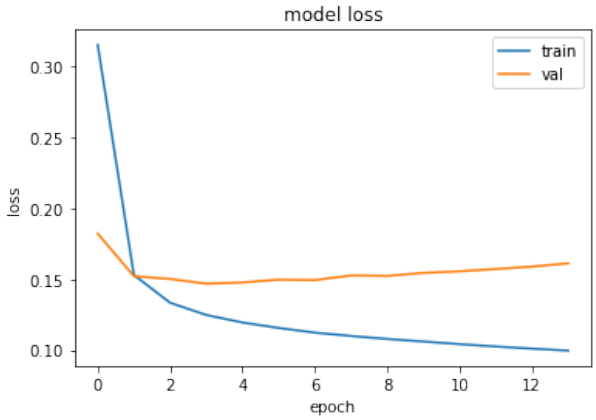

Figura 7.7: Modelo 7

Modelo 8

En el **siguiente modelo**, ha habido un aumento ligero en las neuronas de la capa oculta pero sin ninguna mejora en el modelo. Obtenemos un error de 0.1615, incluso mayor que el anterior y una precisión similar. 7.8 demuestra el modelo inservible y una gran similitud con el anterior.

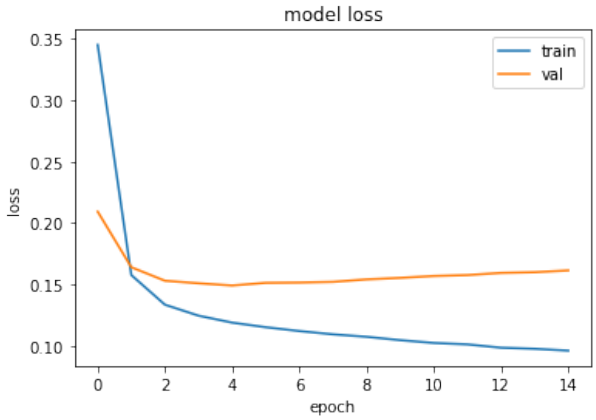

Figura 7.8: Modelo 8

Modelo 9

En **última instancia**, se multiplica por dos el número de neuronas de la capa oculta, obteniendo así un total de 76 neuronas.

Conseguimos disminuir el error y aumentar la precisión al elevar la complejidad de nuestra red, logrando así 0.1235 de error y 97.15 % de precisión. La línea del conjunto de entrenamiento sigue la misma tendencia con el *overfitting* mientras la línea de validación no se nota ningún progreso desde la primera época.

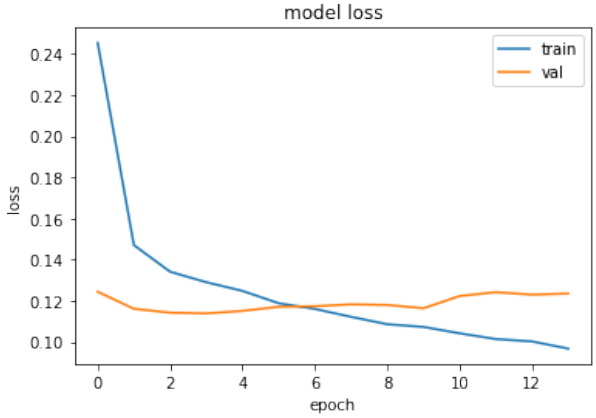

Figura 7.9: Modelo 9

Podemos concluir que la función de coste *Binary Cross-Entropy* no es de gran utilidad para el entrenamiento del modelo que queremos lograr. Los errores son hasta cinco veces superiores a los estudiados con las funciones MAE y MSE. Otro parámetro interesante es las pocas épocas de entrenamiento. Podemos entender que teniendo una *patience* de 10 épocas en el callback, el modelo no consigue mejorar más de 5 épocas seguidas.Se contempla una tabla resumen en 7.6.

Nº Neuronas	**Estructura**	**F. Coste**	**F. Activación**	**val_loss**	**Epochs**
$\frac{in+out}{2} = 20$	[38,20,1]	Binary Cross-Entropy	ReLu	0.1611	14
$\frac{2}{3} \cdot (in + out) = 26$	[38,26,1]	Binary Cross-Entropy	ReLu	0.1615	15
$< 2 \cdot in = 75$	[38,75,1]	Binary Cross-Entropy	ReLu	0.1235	14

Tabla 7.6: Tabla comparación modelos Binary Cross-Entropy

7.1.2. Modelo con dos capas oculta

Para un problema sencillo se ha demostrado que con una capa oculta es suficiente. Aún sabiendo esto observamos en este apartado si el modelo mejora al aumentar el número de capas ocultas.

Por simplificar y no ser repetitivo con las gráficas se expondrá únicamente el mejor modelo probado. El modelo siguiente dispone de una estructura [38,26,26,1] con función ReLu en las dos capas ocultas.

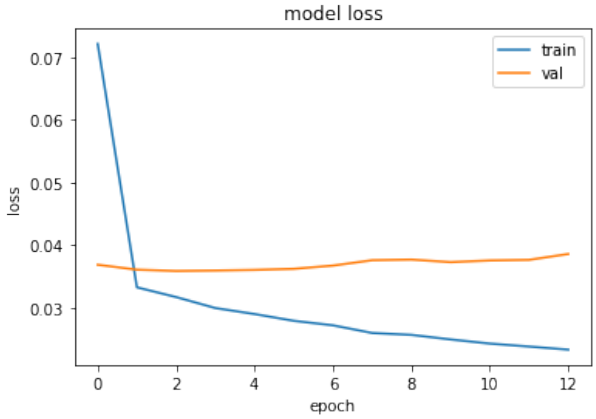

Figura 7.10: Modelo dos capas ocultas

La gráfica 7.10 nos ayuda a concluir que el aumento de capas ocultas no sólo no mejora el modelo, si no que empeora aumentando el error. A parte de las consecuencias de tener una mayor complejidad en la red como es el coste computacional.

7.1.3. Modelo final elegido

Tras la extensa prueba de todos los modelos alternando los hiperparámetros explicados, elegimos quedarnos con el de menor error MSE. Los parámetros de este modelo quedan expuestos en la siguiente tabla.

Parámetros	
Nº Capas	3
Estructura	[38,26,1]
F. Activación	ReLu + Sigmoid
F. Coste	MSE
Callbacks	Early Stopping
Optimizador	Adam

Tabla 7.7: Parámetros modelo escogido

El modelo escogido dispone de una capa oculta. Tiene 36 neuronas en la primera capa de input y 26 neuronas en la capa oculta. Las funciones de activación empleadas son ReLu para la capa oculta y Sigmoid para la final. La función de coste que ha proporcionado el menor error es MSE. Empleamos un Early Stopping que actuó a las 19 épocas.

En la figura 7.11 se observa la evolución del entrenamiento ya comentada en apartados anteriores. El error resultante en el conjunto de validación es de 0.0308. La precisión con la que predicimos el eventro trombótico de un paciente ingresado es del 96.69 %.

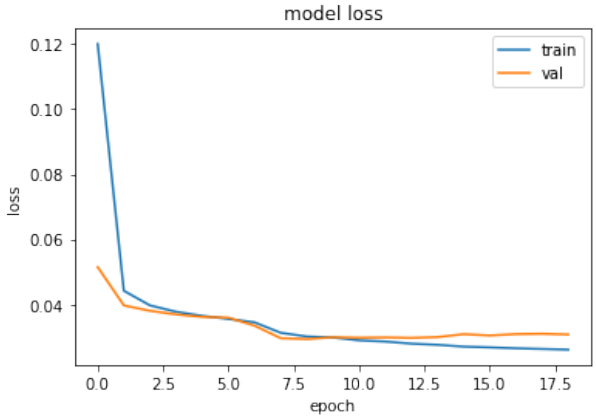

Figura 7.11: Modelo final

8. CONCLUSIONES

En este trabajo de fin de grado se han tratado de lograr ciertos objetivos. El principal ha sido encontrar un modelo de redes neuronales que consiga predecir el desarrollo de un evento trombótico en una persona hospitalizada por COVID-19.

Esta idea parte de un *paper* presentado por el servicio de cardiología del Hospital Universitario La Paz. En éste, se busca mediante varios parámetros estadísticos obtener alguna relación con los eventos trombóticos y la enfermedad por coronavirus. Su resultado no fue exitoso y no se consiguió encontrar un parámetro estadístico que pudiese predecir bien el desarrollos de los eventos tromboembólicos. Si bien, se encontró una relación con la mortalidad y el virus.

Tras nuestro trabajo realizado, se ha logrado el objetivo. Partiendo de la misma base de datos y empleando la inteligencia artificial, se ha logrado encontrar un modelo con un error muy pequeño para predecir los eventos trombóticos.

Existen ciertas diferencias de un trabajo a otro. No se ha empleado exactamente la misma base de datos. Para poder emplear una red neuronal hemos tenido que eliminar una cantidad considerable de variables y algunas decenas de pacientes. La metodología a seguir no ha sido igual ya que las técnicas para la predicción de eventos fueron distintas.

El proyecto propone un modelo de red neuronal artificial compuesto de tres capas. La primera con una dimensión de 38 neuronas que son las variables introducidas a la red. La capa oculta dispone de 26 neuronas con una función de activación ReLu. Por última tenemos una neurona que clasifica los resultados según una función Sigmoid. El optimizador usado para el entrenamiento de la red ha sido Adam.

Nº Neuronas	**Estructura**	**F. Coste**	**F. Activación**	**val_loss**	**Epochs**
$\frac{2}{3} \cdot (in + out) = 26$	[38,26,1]	MSE	ReLu	0.0308	19

Tabla 8.1: Resumen modelo TFG

Con lo expuesto, hemos encontrado un modelo que nos consiga predecir con una precisión del 96.7 % el desarrollo de un evento trombótico en un paciente hospitalizado por la enfermedad del COVID-19. El error *Mean Squared Error* fue de 0.0308.

Por otro lado, en este TFG se han logrado otros objetivos propuestos al principio del mismo. Se ha conseguido profundizar de una manera exhaustiva en la inteligencia artificial, más en concreto, en las redes neuronales. También nos hemos informado de la diversos aspectos de la medicina y hemos hablado con expertos en la materia para aumentar el conocimiento para la posible realización del trabajo. Hemos aportado al estado del arte de la medicina con al Machine Learning y las posibilidades que este campo tiene.

9. LÍNEAS FUTURAS

En el trabajo expuesto hemos tratado dos temas que están en constante evolución. Por un lado el inmenso campo de la medicina está en continuo cambio y adaptación. La tecnología empleada por los profesionales es cada vez mejor y los síntomas en las enfermedades se ven reducidos con el paso del tiempo. Es un campo muy demandante y exigente por parte del trabajador.

En las redes neuronales queda obvio su futuro. Están en nuestro día a día y cada vez aparecen más aplicaciones. Están ayudando en multitud de trabajos, incluso sustituyendo a algunos.

Durante este trabajo ha quedado una cosa clara, la medicina y la inteligencia artificial tienen mucho recorrido. La posibilidad de analizar millones de datos históricos abre un mundo de posibilidades para todas las especialidades en la medicina, ya sean cardiología, neurología, dermatología, odontología, etc. Aunque la experiencia de un buen profesional es difícilmente superable, puede ser complementada. Ya existen diversas aplicaciones implementadas en los hospitales pero pensamos que en un tiempo la cantidad se multiplicará considerablemente.

Respecto a nuestro objetivo en este trabajo se puede mejorar de una manera muy sencilla, mejorando los datos. La base de datos resultó muy escueta y escasa. De meter un mayor número de datos a la red, mejoraría la precisión del modelo. Al aumentar el número de variables, obtendríamos resultados diferentes más precisos.

10. RESPONSABILIDAD SOCIAL E IMPACTO AMBIENTAL

El COVID-19 es una enfermedad que ha afectado a caso 700 millones de personas por todo el mundo. Este número crece diariamente. Surge una necesidad de aportar al estado del arte de la enfermedad para paliar los síntomas y evitar que el número de 7 millones de muertes aumente.

Debido a esta necesidad, empleamos las redes neuronales en el trabajo para ayudar a los profesionales predecir el futuro desarrollo de eventos trombóticos en un paciente ingresado por la enfermedad. La inteligencia artificial puede servir de gran ayuda a los médicos para empezar antes con tratamientos anticoagulantes que prevengan el desarrollo de un evento trombótico grave. Esta medida reduciría la cantidad de muertes provocadas por el virus.

Por otro lado el impacto ambiental es mínimo en la inteligencia artificial. Si bien, emplean electricidad para el entrenamiento y uso de las redes neuronales, no es significativa comparando con las ventajas que puedan aportar.

11. PLANIFICACIÓN Y PRESUPUESTO

11.1. Planificación temporal

En el trabajo se han empleado numerosas horas para la composición del mismo. En este apartado expondremos como se han sido divididos los recursos para llevarlo acabo.

1. **Investigación:** El trabajo tiene una importante parte de investigación. Se quería relacionar las redes neuronales con la medicina desde el primer momento. Aquí comienza la búsqueda por un tema interesante de estudio. Hablando con médicos surgió la idea del COVID-19. La predicción de eventos tromboembólicos fue el tema a tratar de investigar con la ayuda de las redes neuronales. Se han leído numerosos *papers* ya publicados y desde el principio se han ido anotando documentos y referencias para la bibliografía. Tras esta investigación se concluyó la existencia de un gran potencial al trabajo. En esta suma incluimos las horas en reuniones con el tutor. Esta parte ocupó un total de 30 horas.

2. **Formación:** Para la formación han sido clave los cursos online en la plataforma de *Udemy*. Estos cursos han contenido lecciones sobre el Deep Learning, los tipos de redes neuronales y lo más importante, su aplicación en Python. También han sido necesaria la lectura de libros y páginas webs sobre medicina para informarse más al respecto de los tipos de trombos que pueden surgir en el ser humano. La formación ha sido un total de 80 horas.

3. **Preparación datos:** En la depuración de los datos hemos invertido 30 horas. Es la parte más laboriosa en Python ya que los datos a alimentar tienen que tener una estructura determinada. La base en bruto contenía casi 500 variables, muchas inservibles, de las cuales hemos filtrado para finalmente quedarnos con 39.

4. **Modelos y resultados:** El entrenamiento y selección de parámetros de la red neuronal se podría otorgar como la parte más importante del trabajo. Obtener un buen modelo que consiga resolver el problema propuesto tardó 35 horas.

5. **Redacción memoria:** Una vez obtenidos los resultados y análisis que queremos plasmar del trabajo procedemos a la redacción de la memoria. Se ha hecho en la plataforma *Overleaf* con Latex. Esta herramienta ha requerido un aprendizaje notable para su correcto uso. La redacción de la memoria ha sido la parte más extensa en cuanto a horas del trabajo. Han sido 140 horas empleadas en la redacción.

Se han empleado un total de 335 horas para la realización del TFG.

11.2. Presupuesto

En la presente sección llevaremos a cabo una estimación del coste económico que ha supuesto el Trabajo de Fin de Grado. Diferenciamos varios tipos de coste, los materiales y los de mano de obra.

En costes materiales tenemos en cuenta el equipo usado, las licencias utilizadas y algunos gastos indirectos como los gastos en electricidad. Tenemos en cuenta la amortización de los equipos.

Recurso	**Importe (€)**	**Amortización (%)**	**Coste (€)**
Ordenador	1500	20	300
Monitor	300	20	60
Overleaf premium	19	-	19
VS Studio Code	0	-	0
Gasto eléctrico	230	-	230
	TOTAL		**609**

Tabla 11.1: Costes materiales

El cálculo de mano de obra lo haremos teniendo en cuenta un coste de 30 €/hora por parte del tutor y se considera al alumno un valor de 15€/hora. Quedan expuestos la cantidad de horas dedicadas por parte del personal.

Persona	**Coste horario (€/hora)**	**Horas dedicadas (h)**	**Coste total (€)**
Estudiante	15	335	5025
Tutor	30	30	900
	TOTAL		**5925**

Tabla 11.2: Costes mano de obra

Por último realizamos la suma de los dos apartados.

Concepto	**Importe (€)**
Costes materiales	609
Costes mano de obra	5925
TOTAL	**6534**

Tabla 11.3: Costes totales

12. CÓDIGO

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
import keras
from keras.models import Sequential
from keras.layers import Dense
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import confusion_matrix
import tensorflow as tf

# seleccionar las columnas que quiero usar como variables de entrada y como variable objetivo
X = df.drop(columns='trombo')
y = df['trombo']

df = df.sample(frac=1)
sc = StandardScaler()
X = sc.fit_transform(X)

# Definimos el modelo
model = Sequential()
model.add(Dense(26, input_dim=38, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

# Compilamos el modelo, especificando la función de pérdida y el optimizador
model.compile(loss='mse', optimizer='adam', metrics=['accuracy'])

# Entrenamos el modelo utilizando los datos de entrada y salida
callback = tf.keras.callbacks.EarlyStopping(monitor='val_loss', patience=10)
history = model.fit(X, y,validation_split=0.3,epochs=50, batch_size=10,callbacks=[callback])

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc='upper right')
plt.show()
```

Código 12.1: Código utilizado para la creación del modelo

BIBLIOGRAFÍA

«Activation Functions in Neural Networks [12 Types Use Cases]» (s.f.). En: (). URL: https://www.v7labs.com/blog/neural-networks-activation-functions.

Artificial Intelligence: What It Is and How It Is Used (s.f.). URL: https://www.investopedia.com/terms/a/artificial-intelligence-ai.asp.

Caro-Codón, J. et al. (jun. de 2021). «Prediction of thromboembolic events and mortality by the CHADS2and the CHA2DS2-VASc in COVID-19». En: *Europace* 23 (6), págs. 937-947. ISSN: 15322092. DOI: 10.1093/europace/euab015.

«COVID-19 - Wikipedia, la enciclopedia libre» (s.f.). En: (). URL: https://es.wikipedia.org/wiki/COVID-19.

«Deep Learning vs. Machine Learning: Beginner's Guide — Coursera» (s.f.). En: (). URL: https://www.coursera.org/articles/ai-vs-deep-learning-vs-machine-learning-beginners-guide.

Don, S. M., Jayatilake, A. C. y Ganegoda, G. U. (2021). «Involvement of Machine Learning Tools in Healthcare Decision Making». En: DOI: 10.1155/2021/6679512. URL: https://doi.org/10.1155/2021/6679512.

«Gradient Descent Algorithm: A Quick, Simple Introduction — Built In» (s.f.). En: (). URL: https://builtin.com/data-science/gradient-descent.

«Machine Learning: definition, types and practical applications - Iberdrola» (s.f.). En: (). URL: https://www.iberdrola.com/innovation/machine-learning-automatic-learning.

May, M. (ene. de 2021). «Eight ways machine learning is assisting medicine». En: *Nature medicine* 27 (1), págs. 2-3. ISSN: 1546170X. DOI: 10.1038/S41591-020-01197-2.

Neural Networks - Neuron (s.f.). URL: https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/Neuron/index.html.

Pebesma, E. (jul. de 2018). «Simple Features for R: Standardized Support for Spatial Vector Data». En: *The R Journal* 10 (1), págs. 439-446. ISSN: 20734859. DOI: 10.32614/RJ-2018-009.

Rajula, H. S. R., Verlato, G., Manchia, M., Antonucci, N. y Fanos, V. (sep. de 2020). «Comparison of Conventional Statistical Methods with Machine Learning in Medicine: Diagnosis, Drug Development, and Treatment». En: *Medicina* 56 (9), págs. 1-10. ISSN: 16489144. DOI: 10.3390/MEDICINA56090455. URL: /pmc/articles/PMC7560135/%20/pmc/articles/PMC7560135/?report=abstract%20https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7560135/.

Ruder, S. (sep. de 2016). «An overview of gradient descent optimization algorithms». En: DOI: 10.48550/arxiv.1609.04747. URL: https://arxiv.org/abs/1609.04747.

«Top 10 Deep Learning Algorithms You Should Know in 2023» (s.f.). En: (). URL: https://www.simplilearn.com/tutorials/deep-learning-tutorial/deep-learning-algorithm.

«Top 10 Machine Learning Algorithms You Need to Know in 2023 — Simplilearn» (s.f.). En: (). URL: https://www.simplilearn.com/10-algorithms-machine-learning-engineers-need-to-know-article.

«Types of Neural Networks and Definition of Neural Network» (s.f.). En: (). URL: https://www.mygreatlearning.com/blog/types-of-neural-networks/.

«What are Neural Networks? — IBM» (s.f.). En: (). URL: https://www.ibm.com/cloud/learn/neural-networks.

«What Is Deep Learning? — How It Works, Techniques Applications - MATLAB Simulink» (s.f.). En: (). URL: https://www.mathworks.com/discovery/deep-learning.html.

What Is Deep Learning? — How It Works, Techniques Applications - MATLAB Simulink (s.f.). URL: https://www.mathworks.com/discovery/deep-learning.html.

POLITÉCNICA

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES
UNIVERSIDAD POLITÉCNICA DE MADRID

José Gutiérrez Abascal, 2. 28006 Madrid
Tel.: 91 336 3060
info.industriales@upm.es

www.industriales.upm.es