

4 de julio de 2022

Predicción de la contaminación atmosférica mediante redes neuronales

Autor: Ismael Mira Hernández

Tutor: Francisco José García Peñalvo

Cotutor: Roberto López González



**VNiVERSiDAD
D SALAMANCA**

Grado en Ingeniería Informática

Universidad de Salamanca

D. Francisco José García Peñalvo, profesor del Departamento de Informática y Automática de la Universidad de Salamanca.

Certifica:

Que el trabajo titulado “Predicción de la contaminación atmosférica mediante técnicas de inteligencia artificial” que aquí se presenta ha sido realizado satisfactoriamente y bajo su tutela por D. Ismael Mira Hernández para la superación de la asignatura Trabajo de Fin de Grado del Grado en Ingeniería Informática de la Universidad de Salamanca.

Salamanca, 4 de julio de 2022.

D. Francisco José García Peñalvo
Profesor del Departamento de Informática y Automática
Universidad de Salamanca

D. Roberto López González, profesor del Departamento de Informática y Automática de la Universidad de Salamanca.

Certifica:

Que el trabajo titulado “Predicción de la contaminación atmosférica mediante técnicas de inteligencia artificial” que aquí se presenta ha sido realizado satisfactoriamente y bajo su tutela por D. Ismael Mira Hernández para la superación de la asignatura Trabajo de Fin de Grado del Grado en Ingeniería Informática de la Universidad de Salamanca.

Salamanca, 4 de julio de 2022.

D. Roberto López González

Director ejecutivo

Artificial Intelligence Techniques S.L.

Resumen

La contaminación atmosférica es uno de los grandes problemas a los que el mundo contemporáneo tiene que hacer frente. Por ello, existe una creciente necesidad desde, principal pero no exclusivamente, las grandes ciudades de conocer la evolución de los niveles de calidad del aire y poder anticiparse en la toma de decisiones. Para esta tarea, la inteligencia artificial es de gran utilidad en la creación de una estimación que intente mejorar la obtenida por medios tradicionales como modelos teóricos. Concretamente, técnicas de *Deep Learning* y sus aplicaciones como las redes neuronales artificiales, debido entre otras cosas a su capacidad de procesamiento en paralelo, son idóneas para este cometido. En este trabajo vamos a obtener un conjunto de datos sobre la ciudad de Madrid para crear un modelo con el que predecir los valores de diferentes contaminantes a lo largo del tiempo integrándolo en una aplicación web para la visualización de los resultados.

Palabras clave: redes neuronales, contaminación atmosférica, Deep Learning.

Abstract

Air pollution is one of the major problems that the contemporary world has to face. Therefore, there is a growing need from, mainly but not exclusively, large cities to know the evolution of air quality levels and to be able to anticipate decision making. For this task, artificial intelligence is beneficial in creating an estimate that tries to improve those obtained by traditional means such as theoretical models. Specifically, Deep Learning techniques and their applications, such as artificial neural networks, due, among other things, to their parallel processing capacity, are ideal for this task. In this work, we will obtain data about the city of Madrid to create a model to predict the values of different pollutants over time by integrating it into a web application to visualize the results.

Keywords: neural networks, air pollution, Deep Learning.

Índice general

1.	Introducción.....	1
1.1.	Contexto	1
1.2.	Medición de la calidad del aire	1
1.3.	Inteligencia artificial y redes neuronales.....	3
2.	Objetivos.....	4
3.	Conceptos teóricos.....	5
3.1.	Contaminantes principales y límites	5
3.2.	AQI (Air Quality Index).....	7
3.3.	Conjunto de datos.....	8
3.3.1.	Variables en función de su uso	8
3.3.2.	Variables en función de su tipo.....	9
3.4.	Transformación del dataset	9
3.5.	Red neuronal	11
3.6.	Algoritmos de entrenamiento	14
3.7.	Función de coste.....	15
4.	Técnicas y herramientas	17
4.1.	Metodología	17
4.2.	Motor de cálculo.....	18
4.3.	Lenguajes programación	19
5.	Aspectos relevantes	20
5.1.	Procesamiento de los datos históricos.....	20
5.2.	Diseño y pruebas del modelo	23
5.2.1.	Conjunto de datos.....	23
5.2.2.	Arquitectura de la red neuronal.....	32
5.2.3.	Algoritmo de optimización	38
5.2.4.	Función de coste.....	39

5. 2. 5. Validación de los resultados	42
5. 2. 6. Exportación del modelo	48
5. 3. Interfaz web para la visualización de resultados	49
5. 3. 1. Diseño	49
5. 3. 2. Funcionalidad.....	51
6. Conclusiones.....	53
6. 1. Líneas futuras	54
Referencias	55
Anexo técnico	56
Anexo I. Plan del proyecto software.....	56
Anexo II. Especificación de requisitos del software.....	58
Anexo III. Especificación de diseño	58
Anexo IV. Documentación técnica de programación.....	61
Anexo V. Manuales de usuario.....	61

Índice de figuras

Índice de tablas

1. Introducción

1.1. Contexto

Desde la Revolución Industrial, los niveles de contaminación han ido aumentando enormemente, pero no ha sido hasta hace pocos años cuando la sociedad ha comenzado a tomar verdadera conciencia del asunto y los gobiernos han empezado a implantar medidas, quizás viéndose obligados por los malos pronósticos que la comunidad científica auguraba sobre el futuro de nuestro planeta, a bastante corto plazo. Este aumento de la polución se ha reflejado tanto en la aparición de nuevos problemas como en el agravante de algunos existentes, en cuestiones tan diversas como el aumento de las complicaciones respiratorias, la desaparición de especies animales y vegetales, el aumento de la temperatura o la acidificación de los océanos.

1.2. Medición de la calidad del aire

Por esta razón, es fundamental contar con métodos que permitan adelantarse a los episodios de gran contaminación, atenuando sus consecuencias en la medida de lo posible y reduciéndolos a días concretos. En este trabajo se busca construir un modelo que permita conseguir este objetivo, dotando al público general y no solo a las administraciones de una forma de acceder a estos datos, ya que la lucha contra el cambio climático, en cualquiera de sus distintas facetas, es uno de los grandes desafíos a los que el mundo va a tener que afrontar las próximas décadas.

Para la consecución de este desafío, las administraciones públicas deben aplicar normas para disminuir la concentración de sustancias contaminantes en la atmósfera. Esto conlleva un análisis previo de los datos disponibles (recogidos a través de estaciones, similar al proceso seguido con los datos meteorológicos) para garantizar que las decisiones tomadas son las correctas y el grado con el que se está actuando es el adecuado, tanto a corto como a largo plazo. Para la creación de nuestro sistema se utilizarán los datos de estas estaciones, al ser de dominio público. La **figura X** muestra las estaciones existentes en la ciudad de Madrid.

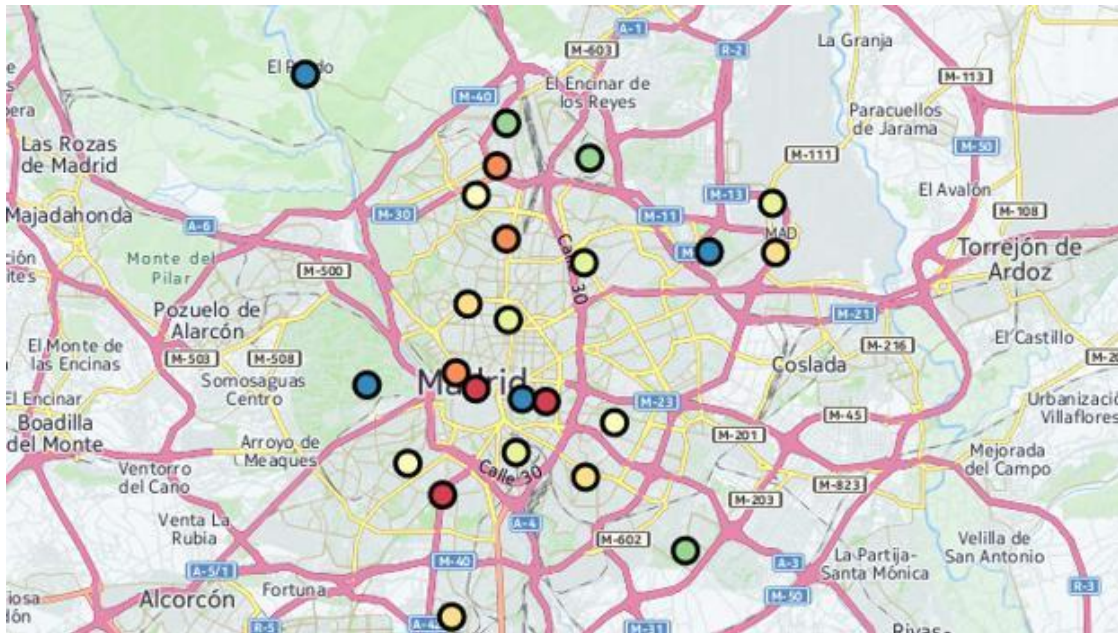


Figura X. Estaciones de medición de la calidad del aire en Madrid.

Desde hace varias décadas existen modelos teóricos que, basándose en la física, intentan simular el comportamiento de un sistema. Existen varios tipos de modelos, como los de dispersión (analizan las emisiones de contaminantes emitidas por una fuente), los fotoquímicos (analizan la efectividad de las estrategias de control frente a la contaminación) o los receptores (identifican y cuantifican las fuentes de emisión de contaminantes).

Usando la capacidad de procesamiento de los computadores actuales además de los avances en inteligencia artificial se busca mejorar estas predicciones. En nuestro caso, aplicando técnicas de Machine Learning y Deep Learning como lo son las redes neuronales, usaremos los datos provistos por las agencias del medio ambiente para crear modelos que permitan predecir con el mínimo error posible los niveles de contaminación de fechas venideras, previniendo episodios de alta contaminación y las consecuencias drásticas que estos conllevan.

1. 3. Inteligencia artificial y redes neuronales

El modelo de nuestro sistema se crea utilizando redes neuronales artificiales. Son una de las más importantes técnicas dentro del mundo de la inteligencia artificial y supusieron una revolución en este campo, difundiéndose enormemente por su capacidad para dar buenos resultados en ámbitos diversos.

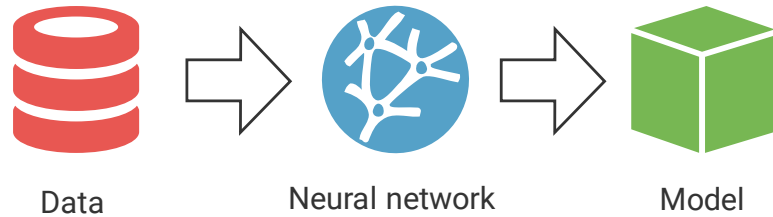


Figura X. Diagrama de actividad de una red neuronal.

Como se puede observar en la figura X, a partir de unos datos de entrada, la red neuronal los procesa y se crea un modelo, de los que hay varios tipos. Los más comunes son los de aproximación, clasificación y predicción. En este caso, por razones obvias, tendremos un modelo de predicción, usados para pronosticar el estado futuro de un sistema a partir de observaciones pasadas sobre él mismo. Otros trabajos de predicción con redes neuronales incluyen pronósticos de ventas, macroeconómicos o de acciones de marketing.

En capítulos posteriores se profundiza en los distintos elementos y aspectos de una red neuronal, ya que será necesario comprenderlos a fondo para la realización de este trabajo.

2. Objetivos

El objetivo principal del trabajo es lograr obtener una predicción fiable del AQI para cada uno de los cinco contaminantes escogidos para la ciudad de Madrid, en un periodo semanal.

Por lo tanto, es una misión fundamental minimizar el error del modelo para conseguir la mayor precisión posible en los valores de salida. Para ello, se construirá la red neuronal adecuada, ajustando correctamente los parámetros y utilizando los algoritmos existentes convenientemente.

Para conseguir el objetivo principal, es imprescindible extraer los datos históricos con exactitud desde fuentes fiables, ya que se utilizarán en el entrenamiento de la red neuronal y las posteriores pruebas. Se realizará un tratamiento de estos para pasar de unos datos sin procesar a unos que se puedan utilizar en el dominio que nos atañe, ordenándolos, encontrando incongruencias y completándolos. Además de ellos, los datos en tiempo real también son necesarios para poder adaptar el modelo lo máximo posible y que así no haya demora en el sistema. Se escogerá la fuente de datos adecuada que nos haga contar con la mayor disponibilidad y sencillez posibles.

Un objetivo secundario de este trabajo es permitir a un usuario sin conocimientos expertos en el tema poder visualizar los valores obtenidos con el modelo de manera sencilla e intuitiva. De esta manera, personas ajenas totalmente a la inteligencia artificial pueden ver la información con una capa de transparencia que oculta toda la parte más tecnológica.

Otro objetivo adicional del trabajo es mejorar a los modelos físicos ya existentes. Se ha demostrado que el Machine Learning puede mejorar a técnicas tradicionales en disciplinas como el análisis de consumidores, el conteo de imágenes o la optimización de tiempos, por lo que puede que en este campo también.

Como último objetivo, además de contar con el modelo para predecir los valores de contaminación para unos datos concretos, se busca integrar lo conocido en el mundo de la inteligencia artificial como “continuous deployment”, prediciendo así los valores de contaminación siempre para la siguiente semana de manera automática, sin tener que realizar tareas manuales.

3. Conceptos teóricos

3.1. Contaminantes principales y límites

Existen gran cantidad de sustancias nocivas presentes en el aire, aunque la EPA (Agencia de Protección Ambiental de Estados Unidos (<https://www.cdc.gov/air/pollutants.htm>)) identifica seis contaminantes como los principales, regulados mediante valores límite basados en los efectos que provocan tanto para la salud pública como para el medio ambiente. Son el monóxido de carbono, el plomo, los óxidos de nitrógeno, el ozono troposférico, la materia particulada y los óxidos de azufre (<https://noticiasonline2020.com/como-afecta-la-topografia-a-la-formacion-del-suelo/>). Se pueden clasificar en primarios o secundarios dependiendo si son generados por procesos humanos/naturales o generados por reacciones químicas de los primarios.

Para este trabajo, vamos a recoger los datos de los que más influencia tienen en Madrid para posteriormente realizar la predicción de cada uno de ellos. El límite marcado para cada uno de ellos viene dado por la Comunidad de Madrid (http://gestiona.madrid.org/azul_internet/html/web/2_3.htm?ESTADO_MENU=2_3), pudiendo ser horarios/diarios (no pueden superarse más de un determinado número de veces al año) o anuales.

Contaminante	Tipo	Límite
PM _{2.5}	Primario y secundario	25 µg/m ³ (media anual)
PM ₁₀	Primario y secundario	40 µg/m ³ (media anual)
O ₃	Secundario	120 µg/m ³ (máxima diaria)
NO ₂	Primario y secundario	40 µg/m ³ (media anual)
SO ₂	Primario	125 µg/m ³ (media diaria)

Tabla X. Tipos de contaminantes y límites para Madrid.

<https://www.comunidad.madrid/servicios/salud/calidad-aire-salud>

- PM_{2.5}: tipo de materia particulada, uno de los indicadores más comunes. Son una mezcla de partículas, tanto sólidas como líquidas, suspendidas en el aire. Estudios realizados en la Unión Europea (<https://www.eea.europa.eu/es/themes/air/intro>) muestran que son el agente contaminante más nocivo para las personas, debido a

la capacidad de penetrar en el cuerpo. En este caso, son las partículas con diámetro menor a 2,5 micras. Son las más dañinas para la salud debido a que pueden llegar al torrente sanguíneo, lo que provoca efectos en el sistema cardiovascular además del respiratorio. Están formadas por elementos más tóxicos procedentes principalmente del tráfico urbano, y son capaces de mantenerse en el aire por más tiempo que las de mayor tamaño.

- PM_{10} : el otro tipo de materia particulada que vamos a medir en este trabajo. Son las partículas con diámetro menor a 10 micras. Pueden ser inhaladas por el sistema respiratorio, aunque no atraviesan los alveolos pulmonares como las anteriormente mencionadas. El tiempo de permanencia es menor, de horas en vez de días. El 77,9% (<https://prtr-es.es/particulas-pm10,15673,11,2007.html>) de emisiones proceden del polvo suspendido existente en la atmósfera.

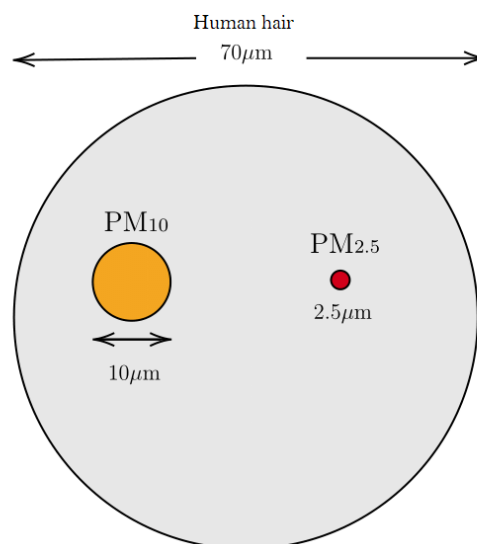


Figura X. Comparación entre tipos de materia particulada.

- O_3 : el ozono troposférico es aquel presente a nivel de suelo. Aunque el ozono es fundamental para proteger de la radiación UV en la estratosfera, en niveles más cercanos la tierra es muy perjudicial. Causa problemas de salud entre los que se encuentran dificultades respiratorias, daño a los pulmones y asma. También tiene efectos adversos sobre la vegetación, interfiriendo con el proceso de la fotosíntesis.

- NO₂: el principal contaminante entre los óxidos de nitrógeno. Producido por fábricas, vehículos y quema de residuos. Es un gas tóxico que además incrementa los niveles de PM_{2.5}. Tiene gran influencia sobre enfermedades respiratorias y provoca la lluvia ácida, con severos efectos sobre la fauna y flora.
- SO₂: el principal contaminante entre los óxidos de azufre y el más peligroso. La mayor fuente de emisión son las fábricas industriales. También incrementan los niveles de materia particulado al igual que los óxidos de nitrógeno. Sobre las personas, afectan principalmente al sistema respiratorio. Sobre el medio ambiente, provocan una disminución en el crecimiento de plantas y árboles.

3. 2. AQI (Air Quality Index)

Es un índice utilizado por agencias gubernamentales para facilitar el acceso a los datos de contaminación y marcar un criterio común, así no teniendo que utilizar las concentraciones reales de los contaminantes y sus respectivos límites. En la Unión Europea no se usa un índice, sino que cada país tiene su propio criterio para comunicar sus niveles, mientras que por ejemplo en Estados Unidos la Agencia de Protección del Medio Ambiente establece una escala de 0 a 500 con un código de colores. (<https://www.airnow.gov/aqi/aqi-basics/>)

Índice de la calidad del aire	Amenaza para la salud	Color
0 a 50	Buena	Verde
51 a 100	Moderada	Amarillo
101 a 150	Insalubre para grupos sensibles	Naranja
151 a 200	Insalubre	Rojo
201 a 300	Muy insalubre	Morado
301 a 500	Peligrosa	Granate

Tabla X. Valores del AQI y significado.

En este trabajo vamos a utilizar valores del índice de la calidad del aire en lugar de valores absolutos, utilizando el AQI estadounidense, tanto en la recogida de datos pasados, actuales y en la propia predicción de los valores futuros.

3.3. Conjunto de datos

<https://www.neuraldesigner.com/learning/tutorials/data-set>

La información necesaria para el entrenamiento de la red neuronal está contenida en lo que se conoce como dataset. El formato más común es CSV, que es el que se usará en este trabajo. Dependiendo del ámbito donde se quiera utilizar, otros formatos como SQL pueden ser más adecuados. Se utiliza un modelo de tabla donde las columnas son las variables mientras que las filas son las muestras.

	Age	Economic Status	Gender	Height	Income	Weight
1	10.465520	Middle Class	Female	139.545894	19663.957892	46.368551
4	43.779882	Rich	Female	127.650202	18274.892352	109.119629
6	35.830860	Middle Class	Female	165.501662	20558.353093	52.728741
7	26.818594	Rich	Female	167.299202	19408.300487	-2.328660
10	31.084558	Middle Class	Female	165.983138	18415.623498	48.371730
11	29.470434	Poor	Female	134.003543	21134.226884	60.585127
12	34.334233	Poor	Female	149.824673	19546.903872	119.229619
14	36.066909	Rich	Female	191.962499	18835.366092	64.921943
17	45.601175	Middle Class	Female	157.798020	18342.649900	32.215115
19	32.509053	Poor	Female	192.663994	20910.754143	52.377219

Figura X. Ejemplo de un conjunto de datos.

Las variables tienen varias clasificaciones posibles. Por un lado, en función de su uso, pueden ser de entrada, de salida o no utilizadas. Por otro lado, en función de su tipo pueden ser numéricas, binarias, categóricas y no utilizadas.

3.3.1. Variables en función de su uso

- Variables de entrada: conocidas como atributos. Desde un punto de vista matemático, son las variables independientes. En nuestro contexto, estas variables de entrada serán series temporales, que representan el histórico de muestras de la ciudad de Madrid.

- Variables de salida: conocidas como etiquetas. Desde un punto de vista matemático, son las variables dependientes. En un problema de predicción como el que se expone en este trabajo, son los valores que queremos obtener, utilizando las variables de entrada necesarias.
- Variables no utilizadas: puede haber columnas que no se usen en la construcción del modelo ya que no aportan información adicional, y en cambio incrementan la complejidad del sistema. Un buen ejemplo son columnas que contengan siempre valores constantes.

3.3.2. Variables en función de su tipo

- Variables numéricas: pueden contener cualquier número tanto positivo como negativo, además de ser decimales. En nuestro sistema son las únicas que usaremos, excepto variables de fecha al ser nuestro dataset una serie temporal.
- Variables binarias: solo pueden tomar dos valores. Normalmente son traducidas a variables numéricas que toman 0 y 1.
- Variables categóricas: similares a las variables binarias, pero pueden tomar más de dos valores. Habitualmente son traducidas a variables numéricas donde cada posible valor es un número.
- Variables de fecha: indican cualquier información relativa a la referencia temporal de la muestra, como puede ser el día, mes, año, día de la semana... Se usarán en la predicción de tendencias y como información adicional.

3.4. Transformación del dataset

Es importante indicar que, debido a que nuestro conjunto de datos son series temporales, es necesario hacer una serie de transformaciones para adaptarlo al entrenamiento de una red neuronal. Esto ocurre debido a que en una muestra del conjunto de datos no existen variables de salida, solamente variables de entrada, por lo que hay que definir qué variables son las que se quieren predecir.

Para este cambio de formato, se introducen los conceptos de lags y steps ahead, que serán dos números. El número de lags indica el número de elementos anteriores que se quiere tener como variable de entrada en una muestra. Para este trabajo, como una

muestra indica un día en concreto, si el número de lags fuera, por ejemplo, 2, significa que en la muestra de hoy aparecen las series temporales de ayer y de antes de ayer.

Por otro lado, el número de steps ahead indica el número de elementos que se van a tener como variables de salida en una muestra. En este trabajo, si por ejemplo solo quisiéramos predecir los valores para el día siguiente el número de steps ahead sería 1.

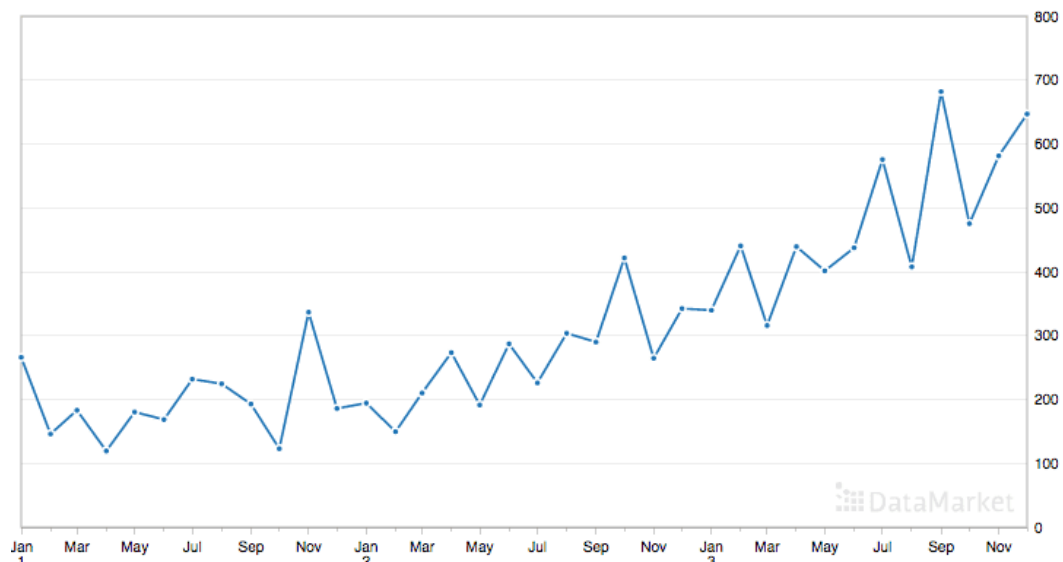


Figura X. Ejemplo de serie temporal.

Por ejemplo, en la figura X, donde el eje horizontal son meses, el lag dos serían dos meses antes del actual, el lag uno es el mes anterior, un step ahead es el mes siguiente y así sucesivamente.

La elección de estos dos valores es importante en el desarrollo correcto de un modelo. Aunque elegir el número de steps ahead es sencillo debido a que representa el número de muestras futuras que se quiere predecir, el de lags no es tan sencillo de escoger ya que no tiene relación directa con los resultados obtenidos, por lo que es necesario realizar pruebas para comprobar qué número de lags es el que más se adecúa al modelo en cuestión a construir.

Por último, el dataset transformado se divide en tres secciones: dataset de entrenamiento, dataset de selección y dataset de validación. El de entrenamiento se usa para probar diferentes modelos con distintas características y así construir el modelo final, que es la función del de selección, ya que elige los que mejores resultados han dado, y por último el de validación prueba el modelo conseguido con muestras no utilizadas en el proceso de entrenamiento con lo que comprueba la validez de este.

Habitualmente, el de entrenamiento corresponde al 60% del dataset completo mientras que el de selección y validación al 20% cada uno. En cualquier caso, para nuestro sistema puede ser que otros porcentajes nos den mejores resultados, por ejemplo, incrementando las muestras usadas en el entrenamiento y validando sólo para el porcentaje correspondiente a 365 muestras, ya que generalmente si se obtienen buenos datos para un año en concreto ocurrirá lo mismo con el resto de los años.

A partir de este momento ya es posible trabajar correctamente con el conjunto de datos procesado, pudiendo realizar tareas que serán útiles en el diseño del modelo como el cálculo de correlaciones entre las variables, el tratamiento de los valores atípicos o el escalado de los datos (introduciendo valores mínimo y máximo).

3. 5. Red neuronal

<https://www.neuraldesigner.com/learning/tutorials/neural-network>

Una red neuronal es un modelo computacional que se inspira en el funcionamiento del cerebro humano. Una arquitectura es una red neuronal con más de una neurona, con parámetros ajustables para cada neurona (pesos y sesgos).

Son de utilidad en un gran número de problemas que se pueden clasificar en tres tipos: aproximación (ajuste de una función a partir de los datos), clasificación (asignar un tipo a partir de unas características) o predicción, como en este trabajo.

Las características de la red neuronal influyen en el buen funcionamiento del modelo, siendo el mayor éxito posible la generalización a otros problemas. En nuestro caso, un ejemplo sería obtener buenos resultados en la ciudad de Barcelona a partir del modelo construido para Madrid. Para ello existen distintas topologías de redes neuronales, que pueden adecuarse más o menos a las características que buscamos, por lo que se entrará en detalle y experimentará con la finalidad de conseguir un error lo más pequeño posible en los valores alcanzados.

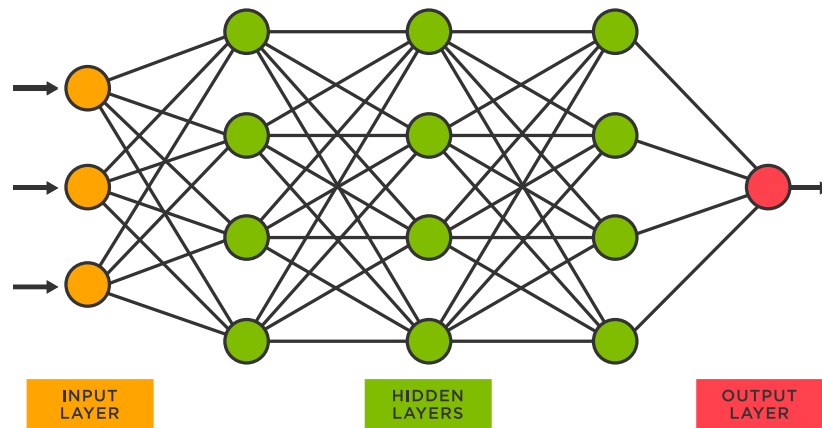


Figura X. Esquema de una red neuronal.

Las neuronas se agrupan en capas, como se puede observar en la figura X, pudiendo estas ser de entrada, ocultas o de salida. Solamente hay una capa de entrada, que es la que recibe los datos, y una de salida, la que genera el modelo. Sin embargo, puede haber varias, una o ninguna capas ocultas. Existen varios tipos de capas.

La más usada universalmente es la capa de perceptrón (también conocido como capa densa). Como particularidad, la entrada se convierte en salida usando una función de activación, teniendo en cuenta unos pesos y sesgos. Existen distintas funciones de activación, como la lineal, la tangente hiperbólica, o la función relu. Los pesos se asocian a una conexión entre dos neuronas y reflejan la intensidad de esta con un valor numérico w_{ij} , siendo i y j dos neuronas. Los sesgos son pesos que se definen para una neurona en concreto con la tarea de mejorar el aprendizaje. En una capa de perceptrón, lo primero que se aplica es una función de combinación (de las entradas con los pesos y sesgos), y después la función de activación para obtener la salida. En la siguiente figura se muestra el funcionamiento.

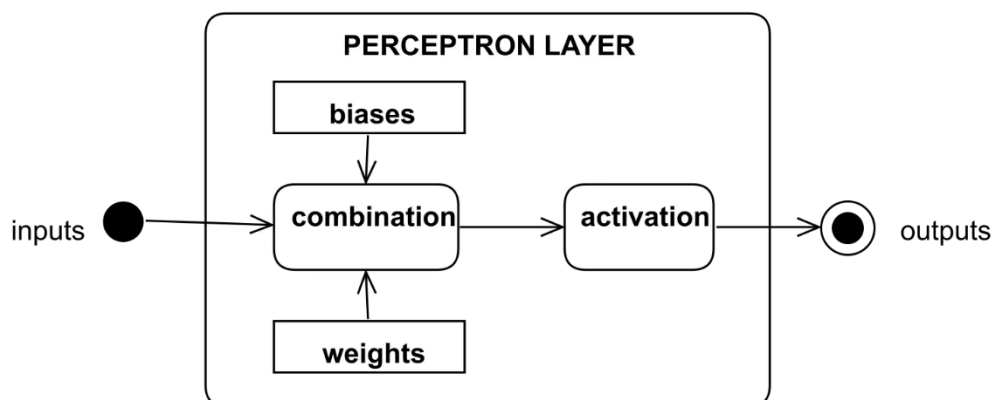


Figura X. Diagrama de una capa de perceptrón.

Otros tipos de capa son la probabilística, usada principalmente en problemas de clasificación por lo que no se entrará en detalle, y la capa LSTM (long-short term memory). Estas son ampliamente usadas en problemas de predicción. Recibe la información en un conjunto de entradas y esta se procesa en distintas “puertas”, conocidas como la de olvido, la de entrada, la de estado y la de salida. La mayor diferencia viene dada porque guarda los estados intermedios en una celda, teniendo cierta memoria, que es lo que le da su nombre a la capa. Son bastante más complejas que la capa de perceptrón y no se pueden tratar de forma matricial como estas otras, pero presentan buenos resultados en conjuntos de datos con series temporales.

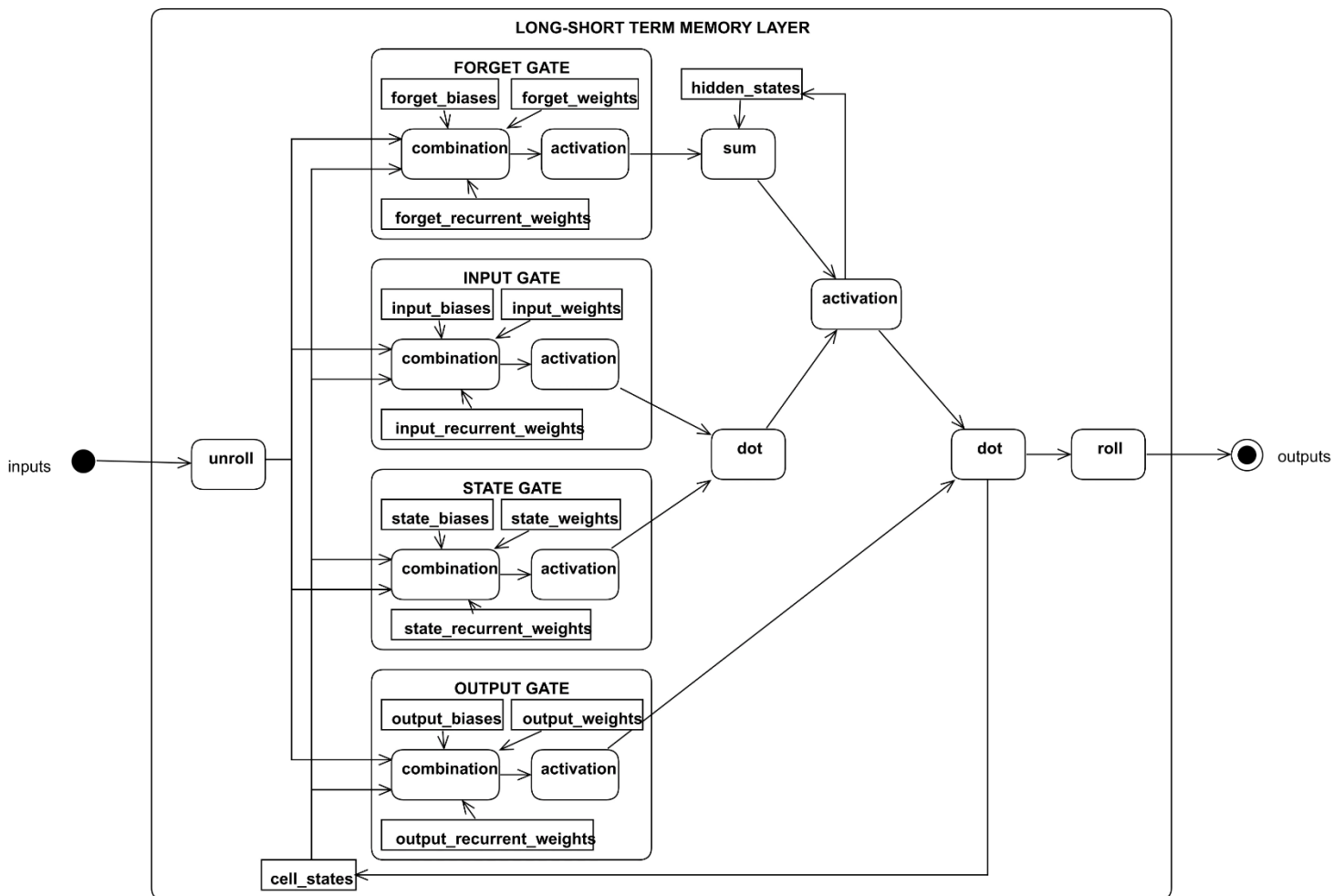


Figura X. Diagrama de una capa LSTM.

Por último, existen otro tipo de capas como las de escalado, desescalado y límite que sirven para tareas concretas como mantener los valores dentro de unos rangos o devolverlos a los rangos iniciales.

Es interesante mencionar que estas capas se pueden combinar, utilizando por ejemplo una capa oculta de perceptrón y otra LSTM. Encontrar la arquitectura adecuada para la red proporcionará mejores resultados.

3. 6. Algoritmos de entrenamiento

https://www.neuraldesigner.com/blog/5_algorithms_to_train_a_neural_network

Este proceso trata de encontrar los parámetros (pesos y sesgos) óptimos para optimizar la red neuronal, y para ello se utiliza un algoritmo de entrenamiento, como se muestra en la figura X.

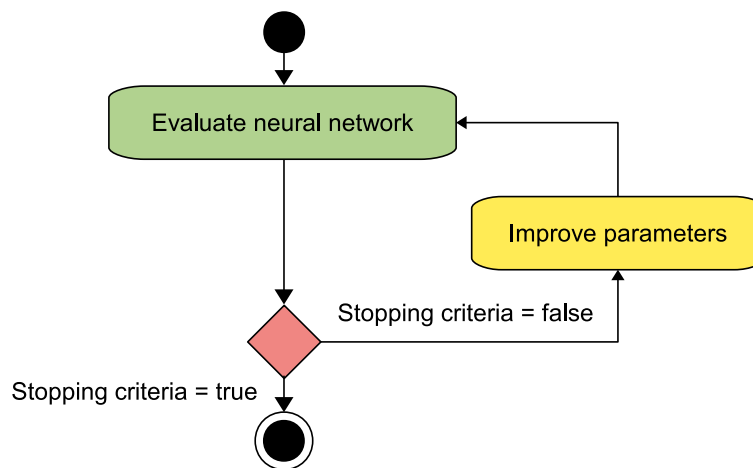


Figura X. Diagrama de actividad de un algoritmo de entrenamiento.

En primer lugar, las muestras del subconjunto de entrenamiento se introducen en la capa de entrada de la red, y se combinan con los pesos y sesgos de esta. Posteriormente se utiliza la función de activación de las neuronas. Acto seguido, se propagan las salidas de la primera capa a la siguiente, y así sucesivamente hasta llegar a la capa de salida. Por último, se calcula el gradiente del error mediante un algoritmo de retropropagación. Si este satisface las condiciones de finalización (que pueden ser un valor mínimo, un tiempo máximo, un número de iteraciones máximo o similares), termina el proceso, en cambio si no lo hace se modifican los parámetros y se repite todo el proceso. Cada repetición del proceso es lo que se conoce como un epoch.

Existen varios algoritmos de optimización, con diferentes características matemáticas y parámetros diferentes para además de minimizar el error, reducir el tiempo de entrenamiento y el número de iteraciones lo máximo posible. Entre los más famosos

destacan el algoritmo de Levenberg-Marquardt, el método de Quasi-Newton y el algoritmo de estimación de momento adaptativo. Normalmente, el mejor algoritmo dependerá del tamaño del conjunto de datos. Para uno pequeño, Levenberg-Marquardt presenta buenos resultados por su rapidez de entrenamiento, aunque utiliza mayor cantidad de memoria que los demás, por eso no es bueno en tamaños mayores. En intermedios, Quasi-Newton trabaja mejor. Por último, en conjuntos de datos muy grandes el algoritmo de estimación del momento adaptativo, siendo el más moderno y adecuado a las nuevas necesidades, funciona mejor que el resto.

3.7. Función de coste

<https://www.neuraldesigner.com/learning/tutorials/training-strategy>

El concepto de función de coste describe la tarea que la red neuronal va a realizar. Permite conocer una estimación sobre la calidad de los resultados obtenidos por un modelo. El objetivo del algoritmo de entrenamiento siempre será minimizar esta función, lo que en nuestro caso significa mejores predicciones.

Esta función se compone de dos términos, el de error y el de regularización. El índice de error es el más importante y un concepto fundamental en este trabajo. Mide el ajuste de la red neuronal al conjunto de datos que se le ha suministrado. En un problema de predicción, indica la diferencia entre las predicciones que ha hecho la red neuronal y los resultados correctos.

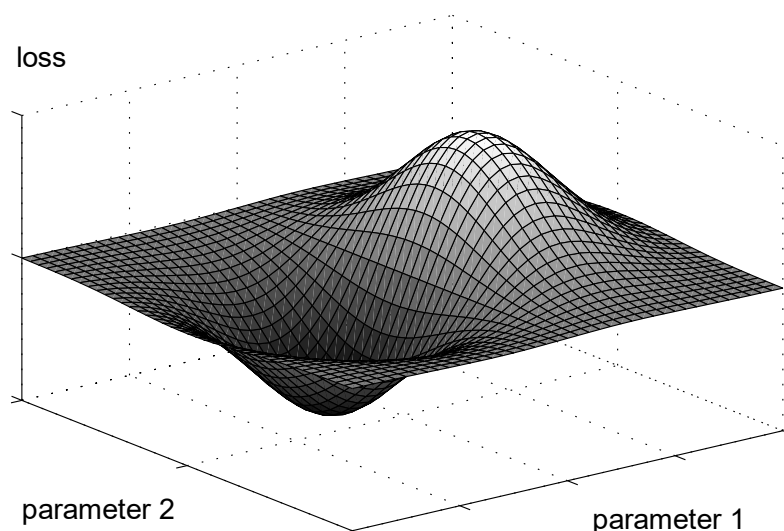


Figura X. Representación gráfica de una función de coste.

Existen diferentes tipos de error, cada uno con ventajas y desventajas.

- La suma de los errores al cuadrado es uno de los más típicos y tiene la ventaja de que puede ser tratado como una función continua diferenciable.
- El error cuadrático medio es similar al anterior pero el error no aumenta con el tamaño del dataset, lo que es útil en conjunto de datos grandes.
- La raíz del error cuadrático medio es la raíz del error mencionado anteriormente.
- El error cuadrático normalizado tampoco aumenta con el tamaño del dataset e introduce un coeficiente de normalización con el que no contaban los anteriores.
- El error de Minkowski evita que el error esté muy marcado por unas pocas muestras con errores muy altos, que es un problema común a todos los anteriormente vistos.

Por otro lado, el término de regularización puede que exista o no, en contraposición al término de error. Un modelo es regular si pequeños cambios en las entradas suponen pequeños cambios en las salidas. Si no fuera así, es irregular por lo que un término de regularización se introduce para controlar la complejidad de la red neuronal. Internamente, hace que los pesos y sesgos sean más pequeños. Existen dos tipos principales:

- Regularización L1: utiliza la suma de los valores absolutos de los parámetros de la red neuronal.
- Regularización L2: utiliza la suma cuadrática de los parámetros de la red neuronal. Se aumenta o disminuye hasta encontrar un balance idóneo.

4. Técnicas y herramientas

4.1. Metodología

Necesitamos una metodología de desarrollo de software que se adapte a las características de este sistema. Debido a ello, metodologías tradicionales, que impliquen una planificación total del trabajo antes de comenzar el desarrollo, no serían óptimas para el problema en cuestión. En este sentido, metodologías ágiles que permitan un desarrollo iterativo e incremental, en vez de lineal, pueden ser de más utilidad para los objetivos de este trabajo.

De esta forma, un marco de trabajo ágil como el que podría ser Scrum, solapando fases del desarrollo (como en nuestro caso podrían ser el diseño de la interfaz web y las pruebas del modelo), con una estrategia de desarrollo incremental y una toma de decisiones a corto plazo, pueden proporcionar un método efectivo para conseguir mayor flexibilidad a cambios, mayor productividad y mejor calidad del software. Sin embargo, la particularidad de este sistema es su enfoque científico, no tan centrado en diseño de software clásico.

Por esta razón, la metodología que se va a usar en este proyecto es DSRM (Design Science Research Methodology), expuesta por Ken Peffers, Tuure Tuunanen, Marcus A. Rothenberger y Samir Chatterjee en <https://www.tandfonline.com/doi/abs/10.2753/MIS0742-1222240302>. Incorpora principios, prácticas y procedimientos necesarios para llevar a cabo una investigación científica y cumplir con tres objetivos principales: ser coherente con la literatura previa, proporcionar un modelo de proceso para hacer investigación en Design Science y proporcionar un modelo mental para presentar y evaluar la investigación de Design Science en un sistema de información como el de este trabajo.

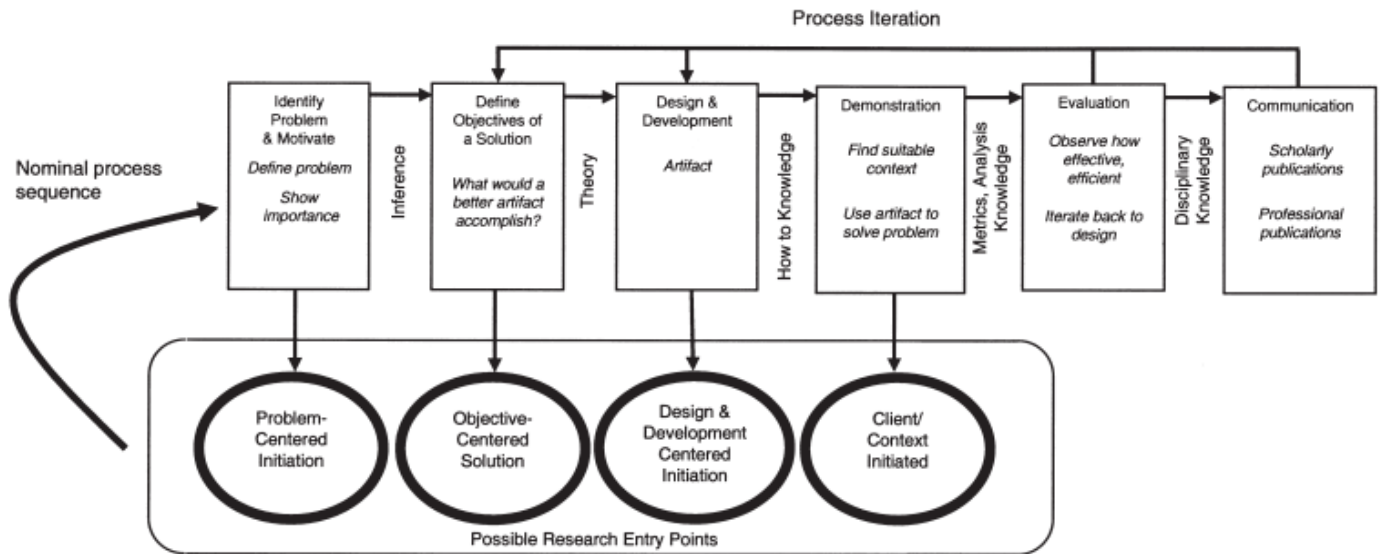


Figura X. Modelo de proceso de DSRM.

Como se observa en la figura X, se incluyen seis pasos a seguir: la identificación del problema y motivación, definición de objetivos para una solución, diseño y desarrollo, demostración, evaluación y comunicación. Asimismo, el proceso permite un desarrollo iterativo acorde con las necesidades mencionadas previamente.

4. 2. Motor de cálculo

El motor de software usado para el diseño de la red neuronal y la obtención del modelo de predicción va a ser la biblioteca de código abierto OpenNN. Esta librería permite construir redes neuronales artificiales con un muy buen rendimiento, obteniendo mejores resultados en aspectos como la velocidad de ejecución y la asignación de memoria que otras bibliotecas de código abierto relacionadas con la inteligencia artificial como TensorFlow o PyTorch.

Haciendo uso de esta librería, podemos construir redes neuronales ajustando los parámetros expuestos en capítulos anteriores. Adopta algoritmos de entrenamiento e índices de error como los mencionados en el apartado (NUM CONCEPTOS TEORICOS), permitiendo poner en práctica los conceptos teóricos previamente mencionados. Además, permite trabajar tanto con aproximación, clasificación y predicción, que es lo que nos interesa en este trabajo.

4. 3. Lenguajes programación

El modelo creado a partir de la red neuronal, además de la expresión matemática, se definirá en lenguaje C++, simplificando la sintaxis. De igual manera, podría traducirse y adaptarse a cualquier otro lenguaje de programación estructurada, al contar con capas de entrada y salida en un orden determinado.

Para la visualización de los resultados obtenidos a partir del modelo, usamos una interfaz web con Node.js como entorno de ejecución de código abierto y Express como su framework de aplicaciones, facilitando el desarrollo de aplicaciones web basadas en Node. Además, facilita la integración con bibliotecas de creación de gráficos (Charts.js) y de obtención de datos a través de APIs (Axios) que son necesarias para su puesta en marcha.

De esta manera, aprovechando la sencillez de los tres lenguajes básicos de creación de sitios web (HTML, CSS y JavaScript) y esta estructura, permitimos que personas sin conocimientos extensos sobre la inteligencia artificial puedan tener una forma gráfica de interactuar con el sistema, dotándolo de una capa de transparencia que oculta el modelo matemático y se centra en los datos finales.




Figura X. Lenguajes de programación utilizados.

5. Aspectos relevante

5.1. Procesamiento de los datos históricos

El primer paso es obtener los datos de años anteriores sobre contaminación y tiempo atmosférico, que posteriormente se convertirán en el conjunto de datos. La información meteorológica será útil en la búsqueda de correlaciones y patrones, permitiendo un mejor aprendizaje de la red neuronal.

Respecto a los contaminantes, hay que tener en cuenta que usamos AQI en lugar de las concentraciones reales. Existen organismos oficiales que presentan y permiten la obtención de las concentraciones, como la Red de Calidad del Aire de la Comunidad de Madrid  http://gestiona.madrid.org/azul_internet/run/j/AvisosAccion.icm). De esta manera, es posible convertir posteriormente estos valores al AQI utilizando las ecuaciones correspondientes a cada contaminantes (<https://forum.airnowtech.org/t/the-aqi-equation/169>). Sin embargo, existen otras fuentes de datos que directamente presentan el valor del índice en lugar de las concentraciones reales, haciendo innecesaria la conversión.

El servicio utilizado, por lo tanto, es World Air Quality Index (<https://aqicn.org/data-platform/>). Este proyecto recopila los datos recogidos de diversas fuentes, dependiendo de la ciudad y el país, convierte los datos al AQI y presenta gráficas detalladas sobre los valores históricos y actuales. Permite la descarga en formato CSV de la información recopilada desde 2014, por lo que nos ofrece un conjunto válido para crear un modelo correctamente. Además, contiene todos los contaminantes de los que queremos información, por lo que no es necesario depender de varias fuentes. Existen algunos problemas como fechas para las que faltan algunos datos por lo que en el entrenamiento de la red habrá que marcar una política para estos casos.

Por otro lado, para la información meteorológica existen muchas más opciones, tanto de organismos oficiales como proyectos de empresas o particulares. En este trabajo utilizaremos Ogimet (<http://www.ogimet.com/index.phtml>), un proyecto personal de G. Ballester Valor que utiliza datos disponibles de forma pública, principalmente de la National Oceanic and Atmospheric Administration del gobierno estadounidense, y los pone al acceso del público. Contiene datos de fechas anteriores a 2014 por lo que podemos usarlo. Un problema es la inexistencia de una opción para descargar los datos, ya sea en

CSV o cualquier otra manera, por lo que es necesario convertirlos a nuestro formato. La información ofrecida es la que se visualiza en la figura X.

Resumen diario a las 13:00 UTC. (12:45 tiempo solar medio)																	
Periodo: 30 días desde 2022/05/01																	
Fecha	Temperatura (C)			Td Med (C)	Hr. Med (%)	Viento (km/h)		Pres. n. mar (Hp)	Prec. (mm)	Nub Tot Oct	Nub baj Oct	Sol D-1 (h)	Vis Km	Diario meteorológico			
	Max	Min	Med			Dir.	Vel.										
01/05	24.6	9.3	17.1	9.8	64.5	S	7.6	1018.9	0.0	4.8	1.8	5.8	23.2				
30/04	23.2	10.2	16.8	8.6	61.3	SSW	5.6	1022.2	0.0	6.0	0.4	11.4	27.0				
29/04	20.9	7.3	13.4	7.9	70.5	N	9.1	1024.0	0.0	3.0	3.3	1.8	28.5				
28/04	18.0	9.5	12.2	8.6	79.7	NNW	6.9	1018.1	1.6	7.6	4.5	1.9	26.5				
27/04	22.1	11.3	14.5	9.9	77.8	S	8.4	1014.1	2.6	7.7	3.6	7.2	23.7				
26/04	20.9	8.1	14.3	5.5	58.6	W	7.3	1011.9	0.0	2.6	1.7	12.4	30.0				
25/04	18.8	4.0	11.9	4.2	63.2	SSW	8.2	1012.9	0.0	1.8	1.1	11.5	30.0				
24/04	16.1	4.6	9.9	3.5	65.0	SW	15.4	1009.9	0.3	3.1	1.8	2.1	28.2				
23/04	11.6	5.4	7.6	5.1	85.3	SSW	22.5	1000.0	18.7	7.5	7.3	0.0	22.0				
22/04	17.4	8.7	12.0	4.2	62.2	SSW	18.7	1003.4	1.0	6.6	3.7	8.5	26.4				
21/04	15.7	3.9	9.4	1.1	56.7	NNW	18.7	1009.3	0.0	3.3	1.8	2.8	28.1				
20/04	19.1	4.0	8.7	4.3	77.5	N	21.2	1007.5	13.3	7.6	6.4	2.4	25.1				
19/04	26.1	8.3	17.1	6.5	54.6	NW	8.3	1009.3	0.0	3.5	2.5	11.6	29.2				
18/04	25.3	9.5	18.0	4.7	43.8	NE	7.2	1014.3	0.0	4.8	0.3	9.0	29.8				
17/04	26.4	6.8	17.3	5.2	48.3	N	8.0	1018.1	0.0	1.4	0.4	11.7	28.2				
16/04	24.8	8.0	17.1	6.7	53.8	NW	5.7	1020.7	0.0	0.3	1.0	11.6	28.4				
15/04	22.3	7.3	15.2	7.6	63.4	NNW	6.3	1019.8	0.0	0.7	1.8	11.1	29.1				
14/04	19.9	8.7	14.5	6.1	58.8	NNE	9.8	1015.6	0.0	3.7	2.7	7.2	30.0				
13/04	18.7	3.9	10.1	6.2	79.0	N	7.2	1012.2	0.6	4.5	2.9	3.3	22.2				
12/04	22.9	8.6	14.1	7.3	68.9	S	17.1	1007.5	4.9	6.8	4.5	6.9	26.4				
11/04	22.6	6.1	15.4	7.2	61.6	SSE	13.4	1009.2	0.0	2.5	1.1	6.0	29.0				
10/04	20.8	9.8	13.9	8.9	73.0	S	10.1	1016.1	0.0	5.2	4.3	3.9	29.5				
09/04	16.8	9.6	13.2	7.4	68.8	SW	19.1	1017.3	0.0	5.8	2.7	5.2	30.0				
08/04	19.9	5.4	12.6	5.4	65.1	SSW	14.9	1019.2	0.0	4.1	0.4	11.9	28.6				
07/04	19.0	5.0	12.3	2.8	54.8	SSW	13.3	1017.7	0.0	1.5	0.2	8.5	29.3				
06/04	13.3	4.0	7.5	1.2	66.9	SSW	12.3	1014.8	0.0	6.2	4.6	0.7	27.5				
05/04	13.9	3.3	7.3	-3.3	48.6	N	10.3	1015.2	0.0	6.5	2.1	7.9	29.9				
04/04	11.9	-0.6	5.3	-6.3	45.2	NNE	14.7	1016.7	0.0	3.9	0.8	10.7	30.0				
03/04	11.0	-2.2	4.6	-4.8	53.8	N	12.8	1019.1	0.0	1.5	2.6	11.7	30.0				
02/04	10.5	0.1	5.3	-4.7	50.2	N	16.8	1019.7	0.0	1.9	2.9	9.4	29.9				

Figura X. Ejemplo información meteorológica.

A continuación, es necesario tratar estos datos para poderlos utilizar en el cálculo de una red neuronal adecuada. Partiendo del CSV obtenido con World Quality Index, añadimos las columnas necesarias para contener la información meteorológica. Estas van a ser los datos atmosféricos básicos, como temperatura, precipitaciones, presión atmosférica, velocidad del viento y humedad relativa.

Filtrando desde 2014, obtenemos esos datos en Ogimet y los añadimos al archivo, teniendo en cuenta fechas para los que no hay datos de contaminación y por lo tanto es inútil añadir los meteorológicos.

Por otro lado, es importante añadir referencias temporales para que la red neuronal realice mejores predicciones en sucesos atípicos. Por ejemplo, los domingos bajan los valores de contaminación al existir menor tráfico. Como la meteorología no influye en este caso, se añaden unas columnas de día, mes y día de la semana para actuar ante estas situaciones.

	DATE	DAY	MONTH	WEEKDAY	PM2.5(AQI)	PM10(AQI)	O3(AQI)	NO2(AQI)	...	HUMIDITY(percentage)
1	01/01/2014	1	1	3	34	11	21	22	...	93.7
2	02/01/2014	2	1	4	36	14	23	25	...	91.3
...
2902	31/05/2022	31	5	2	63	40	24	8	...	49.3

Figura X. Dataset antes del tratamiento.

Finalmente, es necesario transformar el dataset como se ha mencionado en el apartado NUMERO TAL. Debido a que se tratan de series temporales, hay que añadir lags y steps ahead a cada muestra para no aislarlas y que así la red neuronal pueda trabajar correctamente con el mismo.

	DAY_lag_1	MONTH_lag_1	WEEKDAY_lag_1	PM2.5(AQI)_lag_1	PM10(AQI)_lag_1	O3(AQI)_lag_1	NO2(AQI)_lag_1	SO2(AQI)_lag_1	...	HUMIDITY(percentage)_ahead_7
1	1	1	3	34	11	21	22	2	...	80.9
2	2	1	4	36	14	23	25	1	...	78.4
...
2894	23	5	1	76	29	39	11	2	...	49.3

Figura X. Dataset tras el tratamiento.

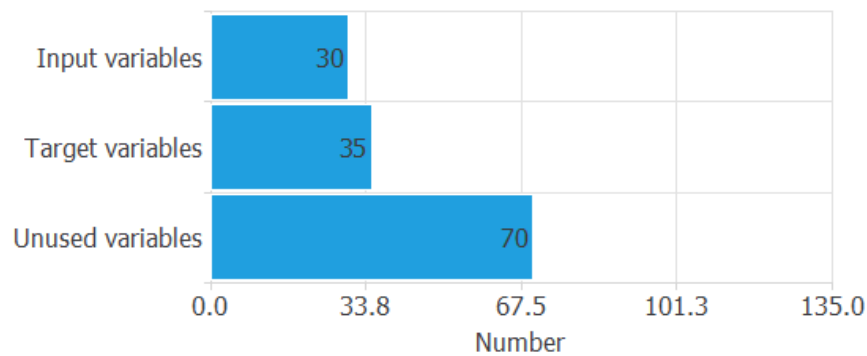


Figura X. Variables según su uso.

5. 2. Diseño y pruebas del modelo

Haciendo uso de los conceptos teóricos y aplicando los datos recabados de otros ejemplos de predicción con redes neuronales podemos diseñar un modelo básico y, a partir del mismo, realizar pruebas para intentar minimizar el error lo máximo posible. Todas las pruebas tienen siete steps ahead (una semana) como aspecto en común. Presumiblemente, el error irá aumentando a lo largo de la semana ya que es más difícil predecir a largo plazo. Como referencia, usaremos el porcentaje de error medio, obtenido para el día 1 y para el día 7.

5. 2. 1. Conjunto de datos

El primer paso consiste en dividir los tres subconjuntos de datos. Podemos aumentar el porcentaje de muestras utilizadas en entrenamiento y reducir las utilizadas en validación a trescientos sesenta y cinco, simulando un año natural y así aportando más información al entrenamiento, en lugar del esquema 60-20-20 típico.

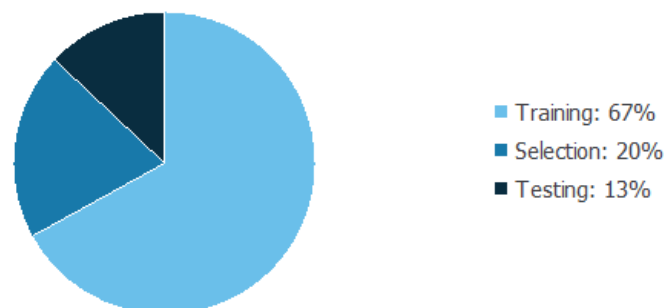


Figura X. Porcentaje de cada subconjunto de datos en el total.

Las estadísticas del conjunto de datos se muestran a continuación. Podemos ver en la figura X los valores máximos, mínimos, medios y la desviación para cada una de las variables del dataset. Con esta información vemos los valores que se alcanzan en los eventos singulares, cuáles son los valores típicos de contaminación y a partir de ello calcular relaciones entre las variables, como se detallará en los siguientes apartados.

	Minimum	Maximum	Mean	Deviation
DAY	1	31	15.7	8.8
MONTH	1	12	6.37	3.49
WEEKDAY	1	7	3.99	2
PM2.5(AQI)	10	166	54.5	19.5
PM10(AQI)	4	344	24.6	13.3
O3(AQI)	1	249	32.8	14.6
NO2(AQI)	2	74	24.2	10.3
SO2(AQI)	0	17	3.11	2.05
PRECIPITATIONS(mm)	0	51	1.07	3.71
TAVG(C)	-3.3	32.9	15.3	8.14
TMAX(C)	0.6	42.7	22	8.94
TMIN(C)	-10.1	24	8.65	6.91
PRESSURE(hPa)	991	1.04e+3	1.02e+3	7.24
WINDSPEED(km/h)	2.5	36.8	10.5	5.43
HUMIDITY(percentage)	16.8	98	58.6	19.5

Figura X. Estadísticas de las variables del conjunto de datos.

Además, en las siguientes figuras se muestra la evolución de cada uno de los cinco contaminantes a lo largo del último año y medio.

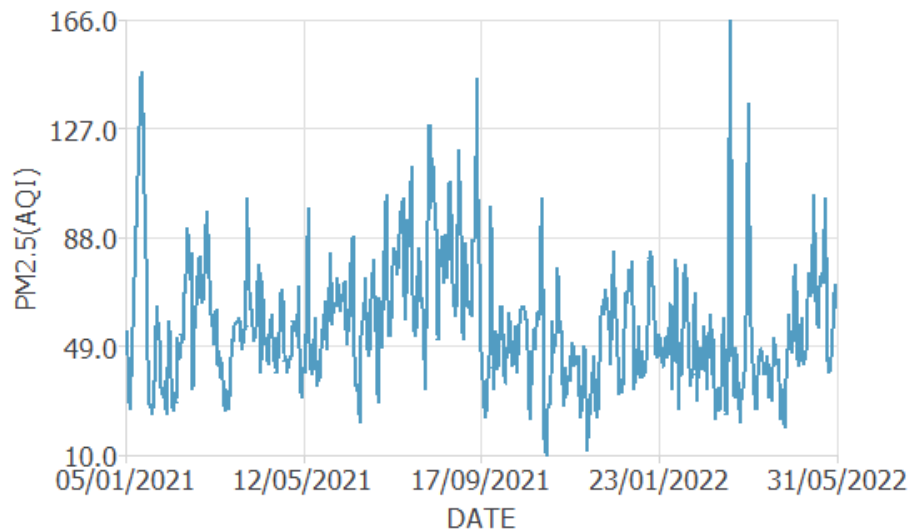


Figura X. Serie temporales del $PM_{2.5}$.

La Figura X muestra la del $PM_{2.5}$, donde observamos la gran irregularidad que existe en los valores de este contaminante, con subidas y bajadas muy pronunciadas además de picos altos.

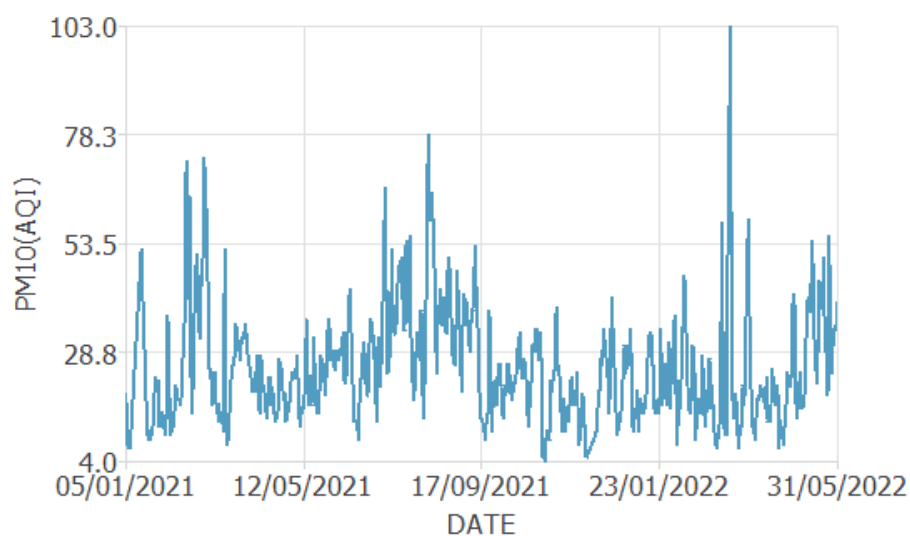


Figura X. Series temporales del PM_{10} .

En la figura X vemos la del PM_{10} . Comparándola con la anterior, vemos la gran relación que tienen los dos tipos de materia particulada, teniendo prácticamente los mismos eventos singulares y una evolución similar a lo largo del periodo.

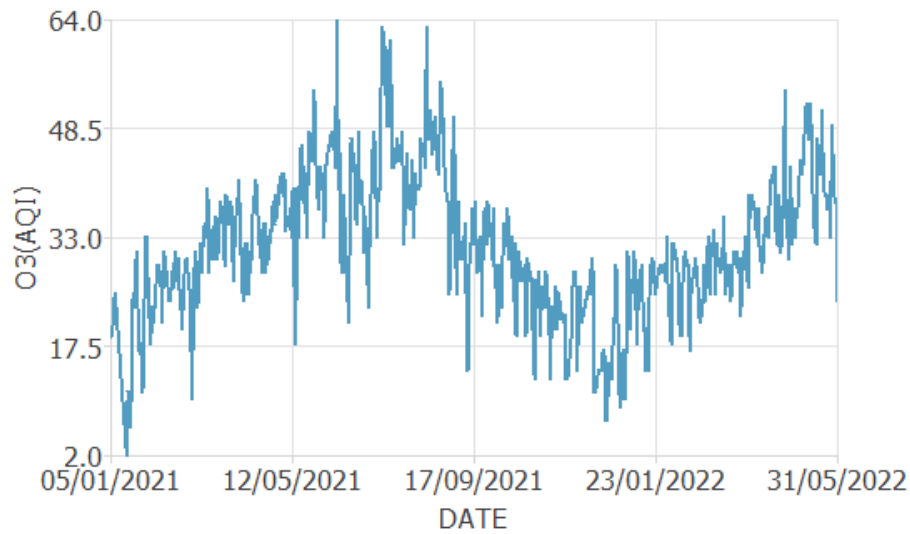


Figura X. Series temporales del O3.

En la figura X vemos la del O3. A partir de la gráfica deducimos que el nivel de ozono troposférico aumenta durante la primavera y el verano, disminuyendo posteriormente en otoño e invierno. Esta hipótesis se corroborará más adelante con las correlaciones.

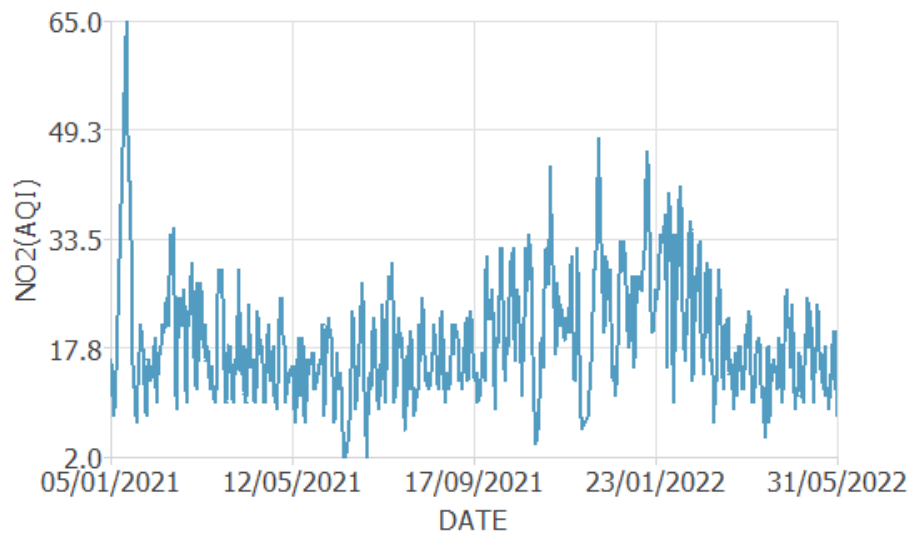


Figura X. Series temporales del NO2.

En la figura X vemos la del NO2. Vemos que los eventos singulares (solo hay uno grande a comienzos de 2021) no están relacionados con los de la materia particulada. Además, los valores para este contaminante son bastante estables, siempre alrededor del mismo rango de valores.

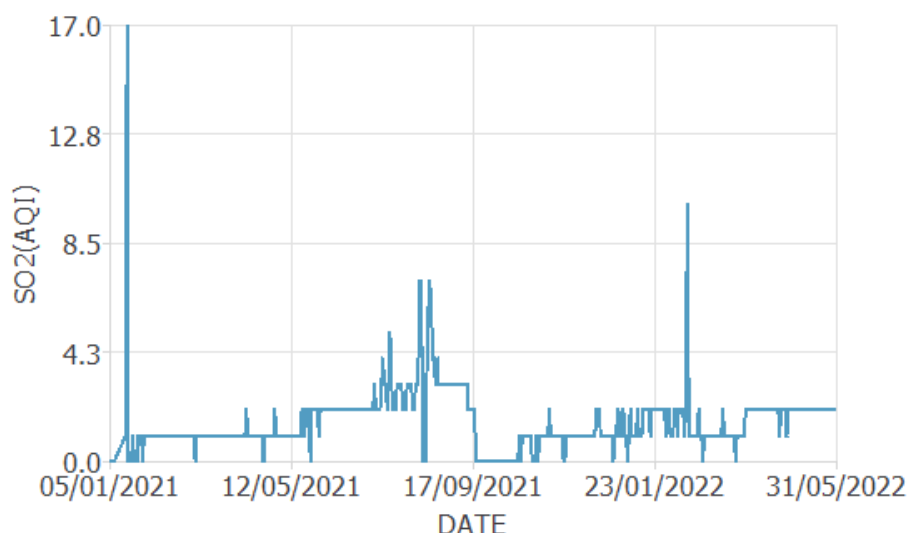


Figura X. Series temporales del SO2.

Por último, en la figura X vemos la del SO2. Se muestran unos niveles muy bajos a lo largo de todo el periodo, con algún evento singular relacionado con los del NO2. Vemos que es el contaminante que menos es identificado en Madrid.

Seguidamente, se van a mostrar las correlaciones cruzadas. Estas indican la relación que tienen dos variables concretas a lo largo del tiempo, y cómo una variable de entrada influye en una de salida. Como nuestro número de variables de entrada es grande, podríamos obtener una cantidad muy elevada de gráficas de este tipo. Debido a ello, se muestran solo un par en las figuras X y X que nos indican la correlación cruzada entre el PM_{2.5} y tanto el PM₁₀ como las precipitaciones.

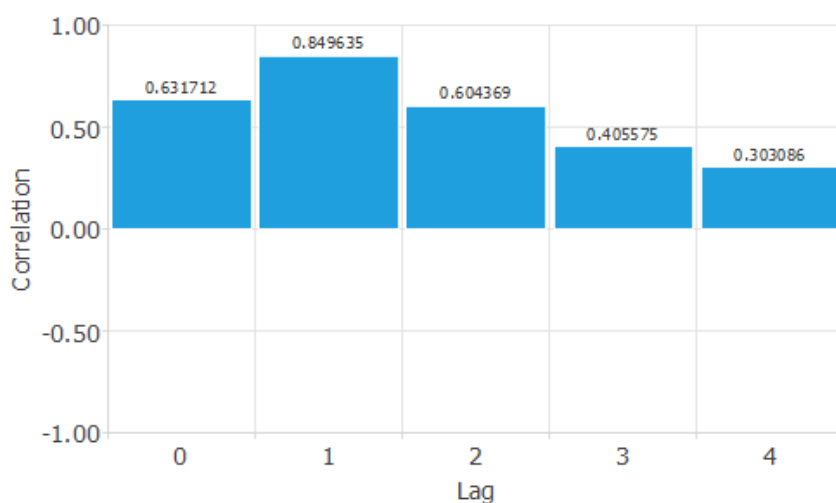


Figura X. Correlación cruzada del PM_{2.5} respecto al PM₁₀.

En la figura X podemos observar los números muy altos para los dos tipos de materia particulada, acercándose mucho a valores cercanos al uno, que significa proporción directa. Por lo tanto, estas dos variables están muy relacionadas entre sí.

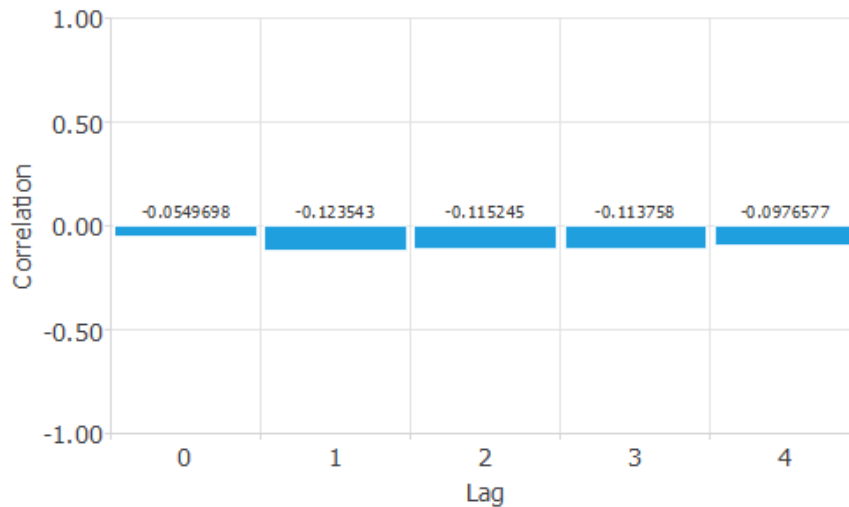


Figura X. Correlación cruzada del PM_{2.5} respecto a las precipitaciones.

En contraposición, en la figura X vemos los pequeños valores que tiene la correlación entre el PM_{2.5} y las precipitaciones, indicando la poca dependencia que tienen estas dos variables entre sí.

Sin embargo, los valores de correlaciones verdaderamente importantes se muestran en las siguientes figuras. Se indican, para cada uno de los cinco contaminantes, las variables que más influencia tienen en la predicción futura de los mismos. Los valores positivos indican que a medida que aumenta la variable de entrada, aumenta la de salida, y los valores negativos indican que a medida que aumenta la variable de entrada, disminuye la de salida.

Podemos observar en todas las figuras el mismo patrón: la variable que más peso tiene en el cálculo de la predicción es el valor del propio contaminante el día anterior. Lógicamente es así, ya que no suelen variar mucho de un día a otro y es obvio pensar que la mayor influencia sea la variable del día pasado.

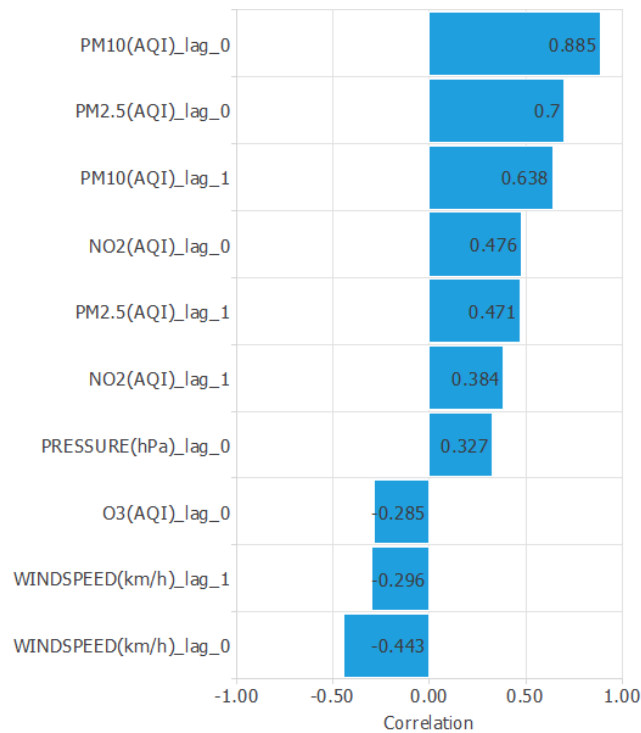


Figura X. Mayores correlaciones para el PM_{2.5}.

En la figura X vemos las del PM_{2.5}. Observamos la gran influencia que tienen el resto de los contaminantes en el aumento de este, y a su vez la gran influencia que tiene la velocidad del viento en el descenso del mismo. El sentido de esta afirmación viene dado por la dispersión que provoca una mayor fuerza del viento. Como se dispersan las partículas, no se concentran en el lugar típico (centro de las ciudades debido al tráfico) sino que llegan a puntos más distantes, por lo que al aumentar la velocidad del viento disminuyen los valores de materia particulada.

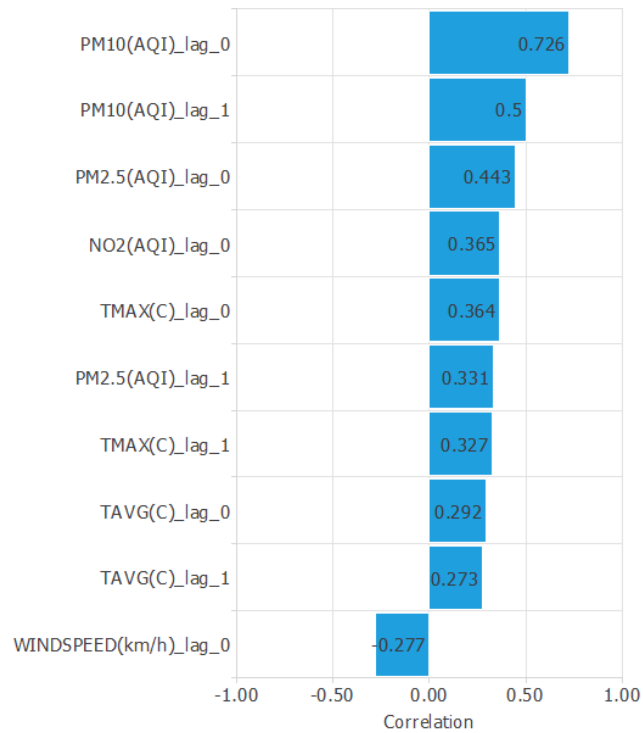


Figura X. Mayores correlaciones para el PM₁₀.

En la figura X observamos que ocurre algo similar para el PM₁₀. El resto de los contaminantes tienen mucha influencia en el aumento de este y el viento en el descenso (aunque no de manera tan marcada como para el PM_{2.5} ya que al pesar más las partículas es necesaria una mayor fuerza del viento).

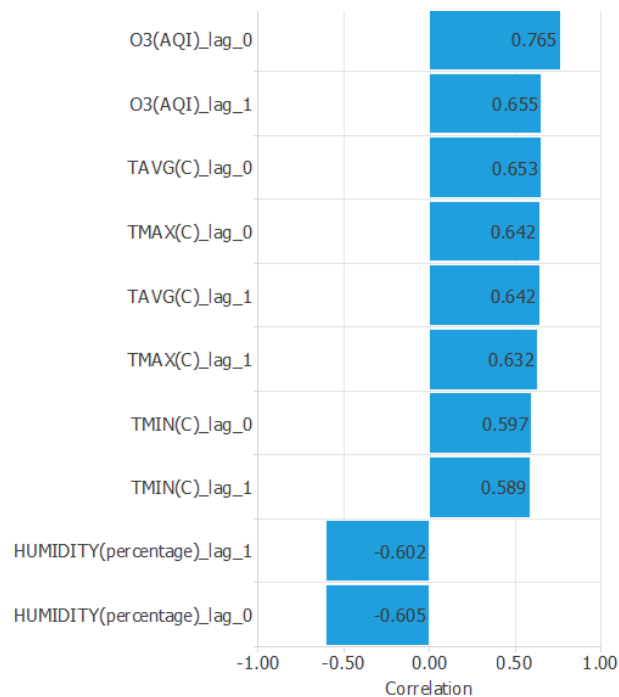


Figura X. Mayores correlaciones para el O₃.

En la figura X vemos las correlaciones del O3, observando la gran importancia que tiene la humedad relativa en la predicción de los valores del ozono troposférico. Podemos deducir, por lo tanto, que los niveles de este contaminante aumentan a lo largo del verano (cuando hay poca humedad relativa) y disminuyen en invierno, como se mencionó anteriormente con la serie temporal del ozono.

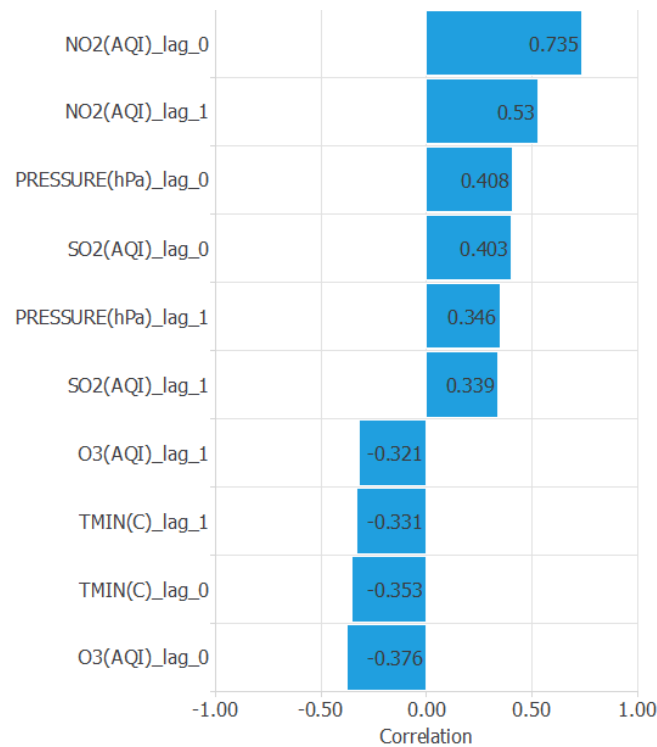


Figura X. Mayores correlaciones para el NO2.

En la figura X vemos las del NO2. Observamos que existe correlación con muchas variables diferentes como la presión atmosférica o la temperatura mínimo. Un dato que destacar es el descenso de los niveles de NO2 si aumenta el O3, por lo tanto, podemos afirmar que en invierno (cuando menores son los valores del O3) es cuando más altos suelen ser los valores de NO2.

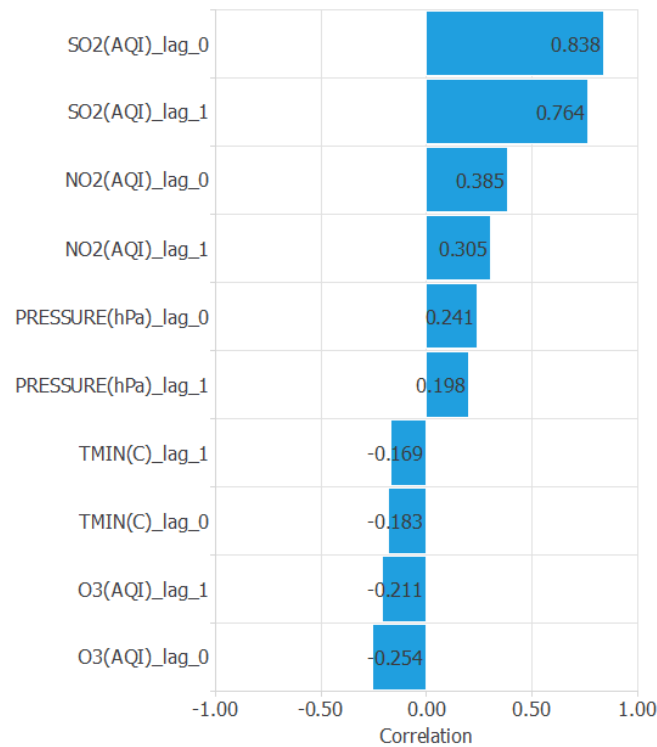


Figura X. Mayores correlaciones para el SO2.

Por último, vemos en la figura X las del SO2. Vemos que las mayores correlaciones, con mucha diferencia, son con los valores de sí mismo de los días anteriores. El resto de las variables, a parte del NO2 (si uno aumenta, el otro también), no influyen drásticamente en el dióxido de azufre.

5. 2. 2. Arquitectura de la red neuronal

Se han probado distintas arquitecturas para la red neuronal aplicando los conceptos teóricos y ejemplos anteriores de predicción con este método. El objetivo es minimizar el error y acercar la predicción lo máximo posible a los valores reales.

Como primera medida de minimización de este, comprobamos el efecto que tiene aumentar el número de lags a más de uno, añadiendo información adicional a cada muestra. En la siguiente tabla se muestra el porcentaje de error (%) para las variables de salida, dependiendo del número de lags.

	2 lags	5 lags	10 lags
PM_{2.5} (1 step ahead)	5,00%	6,54%	5,51%
PM₁₀ (1 step ahead)	2,08%	2,17%	2,08%
O₃ (1 step ahead)	2,10%	2,33%	2,71%
NO₂ (1 step ahead)	7,71%	11,91%	10,84%
SO₂ (1 step ahead)	3,93%	5,56%	4,24%
PM_{2.5} (7 steps ahead)	10,54%	10,35%	10,41%
PM₁₀ (7 steps ahead)	2,92%	2,84%	2,85%
O₃ (7 steps ahead)	2,55%	2,39%	2,73%
NO₂ (7 steps ahead)	12,29%	12,97%	12,32%
SO₂ (7 steps ahead)	7,20%	7,06%	5,73%

Podemos observar que no existe una mejora significativa, incluso empeora en algunos casos, por lo tanto, para no añadir complejidad al dataset transformado nos quedamos con dos lags. Además, vemos que el error aumenta proporcionalmente cuanto más lejos queramos predecir, por lo que el indicador del primer día es suficiente para visualizar el error.

En primer lugar, partimos de una red neuronal con una sola capa de perceptrón con una neurona por cada variable de salida, que es la configuración más simple existente. Como algoritmo de entrenamiento, usaremos el método de Quasi-Newton, que es útil en distintos tipos de aplicaciones y por lo tanto se trata como el algoritmo “por defecto”. A lo largo del entrenamiento, el error de entrenamiento y el de selección irán disminuyendo hasta llegar a un criterio de parada. En la figura X donde vemos las estadísticas, observamos lo rápido que ha sido el entrenamiento.

	Value
Training error	0.598
Selection error	0.657
Epochs number	56
Elapsed time	00:00:02
Stopping criterion	Minimum loss decrease

Figura X. Estadísticas red neuronal con perceptrón.

Como nuestro conjunto de datos es de tamaño medio, una sola capa es insuficiente. Por ello, añadimos una segunda capa de perceptrón utilizando una función de activación sigmoide como puede ser la tangente hiperbólica. Para decidir cuál es el número de neuronas que tiene esta nueva capa, realizamos ensayos comenzando con un número bajo de neuronas como puede ser 3 y aumentando hasta encontrar los mejores resultados.

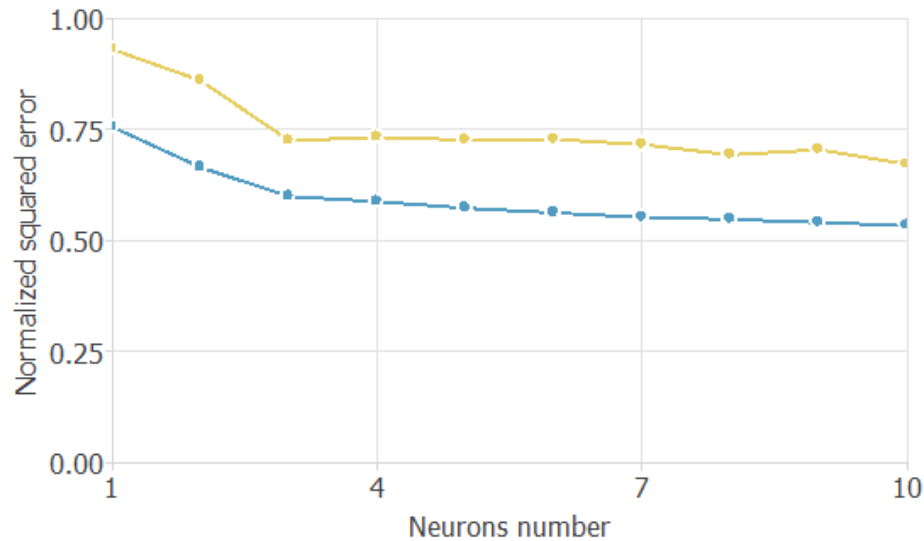


Figura X. Evolución del error dependiendo del número de neuronas

Como se puede observar en la figura X, donde la línea azul es el error de entrenamiento y la amarilla el de selección, diez neuronas proporcionan el número mínimo de error con esta configuración. La red funciona peor con un número menor de neuronas y añade complejidad innecesaria con un número mayor, ya que no mejora los resultados.

Como ensayo, creamos otra tercera capa similar a la anterior, aunque en un dataset no muy grande como este no suele proporcionar una mejora significativa para la complejidad que añade.

	2 capas	3 capas
PM_{2.5}	5,67%	5,58%
PM₁₀	2,07%	2,13%
O₃	2,00%	2,20%
NO₂	8,35%	9,56%
SO₂	3,96%	5,21%

Tabla X. Comparación de errores entre dos y tres capas de perceptrón.

En la tabla X, observamos que la tercera capa mejora levemente el error de $PM_{2.5}$ pero empeora todos los demás, por lo tanto, no es de utilidad. En este caso se prioriza la simplicidad y mejores resultados globales que ofrecen dos capas de perceptrón.

Posteriormente, vamos a comprobar el funcionamiento de las capas LSTM y su posible utilidad para este problema en concreto. Para comenzar, eliminamos la capa de perceptrón adicional que habíamos introducido previamente y añadimos una capa LSTM básica con 3 neuronas. Después, aumentamos el número de neuronas al igual que hicimos con el perceptrón para añadir complejidad. Es importante mencionar que el algoritmo de Levenberg-Marquardt no es válido con este tipo de capas, por lo que utilizaremos Quasi-Newton. En las siguientes figuras vemos los datos del entrenamiento.

	Value
Training error	0.565
Selection error	0.548
Epochs number	218
Elapsed time	00:00:54
Stopping criterion	Maximum selection error increases

Figura X. Estadísticas red neuronal con LSTM.

Vemos que los niveles de error de entrenamiento y selección son ligeramente mejores a la estrategia inicial, en cambio el tiempo de entrenamiento aumenta enormemente. Además, el criterio de parada en este caso ha sido por incremento del error de selección, en vez de disminución de error mínima como en la figura X.

El siguiente paso es combinar una capa LSTM con un perceptrón, haciendo uso de ambas, buscando aprovechar las ventajas que nos ofrecen los distintos tipos.

	Solo perceptrón	Perceptrón + LSTM (3 neuronas)	Perceptrón + LSTM (10 neuronas)
PM_{2.5}	5,67%	9,24%	7,94%
PM₁₀	2,07%	3,00%	2,81%
O₃	2,00%	2,71%	2,24%
NO₂	8,35%	14,92%	9,78%
SO₂	3,96%	5,83%	5,99%

Tabla X. Comparación de errores entre capas de perceptrón y LSTM.

Vemos que ningún caso es mejor que los resultados obtenidos en pruebas anteriores utilizando solamente capas de perceptrón. Esto puede ser debido a que, en un problema donde el número de variables de salida no es extremadamente grande (treinta y cinco) y el número de muestras y columnas tampoco, las capas LSTM añaden una complejidad muy grande e innecesaria. En ejemplos más grandes, como pueden ser los relacionados con el campo de la medicina y biología, con cantidades enormes de muestras (pacientes) y variables (genes), sí que puede ser beneficioso el uso de este tipo de capas, sin embargo, para este trabajo no es lo óptimo. Además, el tiempo de entrenamiento aumenta a unos niveles muy altos, como más de once minutos con diez neuronas.

Tras la realización de estas pruebas, la arquitectura final de la red neuronal se muestra en la figura X.

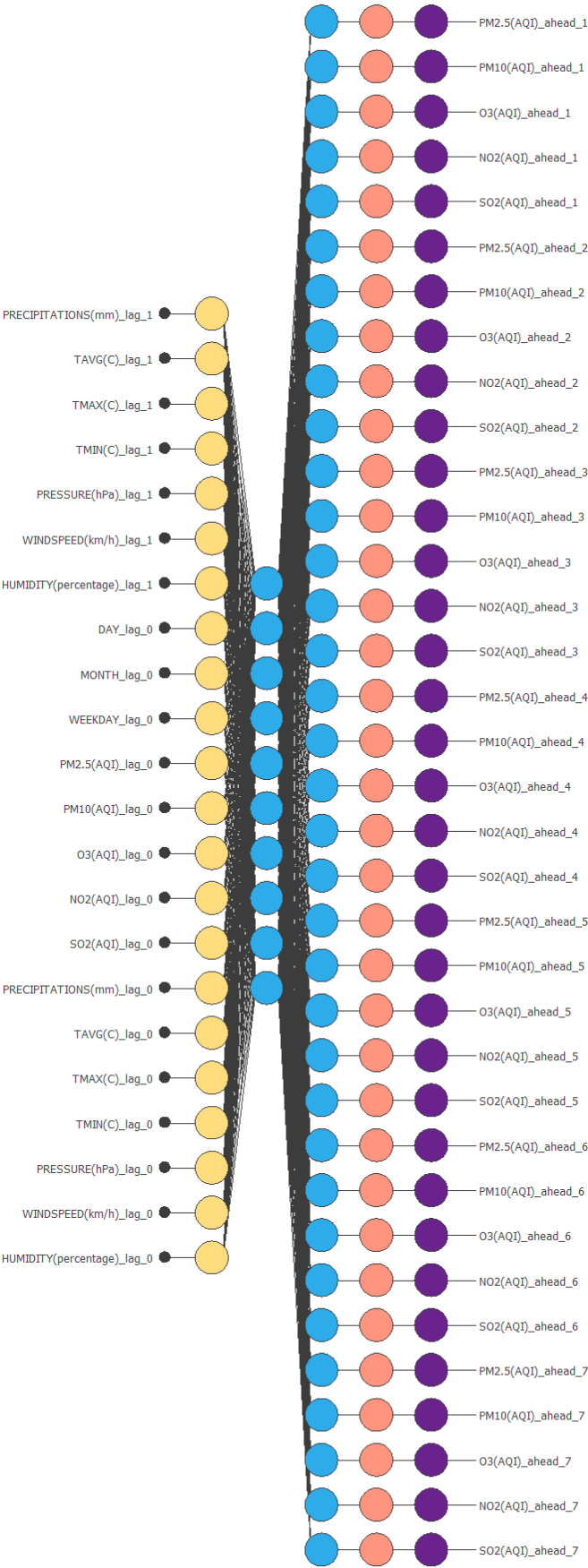


Figura X. Arquitectura de la red neuronal.

La primera capa (círculos amarillos) es una capa de escalado de veintidós neuronas (una por cada variable de entrada).

La segunda capa es una capa de perceptrón (con función de activación tangente hiperbólica) con diez neuronas, ya que se demostró en el capítulo X que era lo óptimo. A continuación, se encuentra otra capa de perceptrón con función de activación lineal y treinta y cinco neuronas (una por cada variable de salida). Estas dos capas de perceptrón se representan con círculos azules.

Por último, se encuentran una capa de desescalado (círculos naranjas) para eliminar el escalado de la primera capa y otra de filtro (círculos morados), que para este trabajo no tiene utilidad alguna y por lo tanto no se usa.

5. 2. 3. Algoritmo de optimización

El siguiente paso es probar otros algoritmos de optimización. La teoría nos dice que para un conjunto de datos relativamente pequeño como este (con alrededor de 10 columnas y unas pocas miles de muestras) el algoritmo de Levenberg-Marquardt es el idóneo. Además, se prueban algoritmos tradicionales basados en el gradiente y el de estimación del momento adaptativo, que en los últimos años ha conseguido mejorar resultados previos. Los datos de los entrenamientos son los siguientes:

	Quasi-Newton	Levenberg- Marquardt	Gradient Descent	Adaptative Moment Estimation
PM_{2.5}	5,67%	5,54%	5,28%	6,74%
PM₁₀	2,07%	2,05%	2,02%	2,27%
O₃	2,00%	2,00%	2,00%	2,01%
NO₂	8,35%	7,87%	7,98%	8,96%
SO₂	3,96%	4,03%	4,48%	4,58%

Tabla X. Comparación de errores entre los algoritmos de optimización.

Podemos ver en la tabla X que Levenberg-Marquardt es en efecto el que mejor resultados globales ofrece, respaldando la teoría, aunque prácticamente similares a Quasi-Newton. El algoritmo de gradiente también funciona bien excepto para el SO₂. Sin embargo, el algoritmo ADAM (estimación del momento adaptativo) es notablemente peor para este conjunto de datos.

5. 2. 4. Función de coste

Lo siguiente que es necesario comprobar es el tipo de error que conviene más en este caso. Hasta ahora, se utilizaba el error cuadrático medio en todos los casos, por eso probaremos los otros tipos.

Cambiándolo por el resto de los mencionados en el apartado NUM APARTADO, obtenemos los siguientes valores. Para poder entenderlos, se añaden las fórmulas del cálculo del error cuadrático normalizado y el de Minkowski, que son los típicamente utilizados en entrenamiento de redes neuronales.

$$normalized_squared_error = \frac{\sum (outputs - targets)^2}{normalization_coefficient}$$

$$minkowski_error = \frac{\sum (outputs - targets)^{minkowski_parameter}}{samples_number}$$

	Training	Selection	Testing
Sum squared error	2149.88	1270.35	1733.69
Mean squared error	1.23486	2.19026	2.98912
Root mean squared error	1.11124	1.47995	1.72891
Normalized squared error	0.437358	0.581669	0.753081
Minkowski error	10654.6	5658.94	6714.42

Figura X. Estadísticas de los índices de errores

Para este trabajo, usaremos el error de Minkowski. Consiste en elevar cada instancia a un número a nuestra elección en vez de siempre al cuadrado, como hace el error cuadrático medio. Se suele utilizar para minimizar el error cuando existen muchos valores atípicos. Sin embargo, utilizando como parámetro de Minkowski un número menor que dos perdemos capacidad de predicción de eventos singulares, que es un aspecto que nos interesa mucho en el tema de la predicción de la contaminación, ya que estos días son los más problemáticos para la salud y el medio ambiente. Los eventos singulares son aquellos en los que hay un día en concreto con valores de contaminación muy superiores a los días anteriores y posteriores. Predecir estos eventos es uno de los grandes desafíos de las redes neuronales y una parte muy compleja en el desarrollo de estas. Un ejemplo de uno de ellos el siguiente.

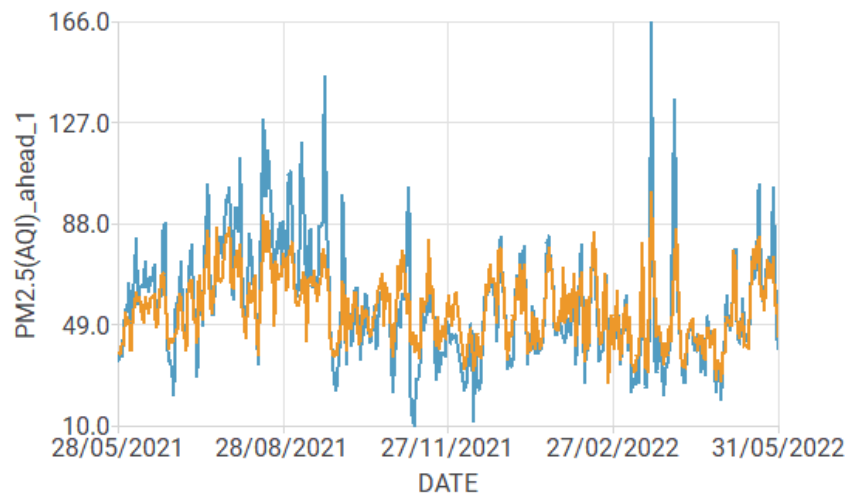


Figura X. Valores reales y predicción del $PM_{2.5}$.

La figura X muestra en azul los valores reales de $PM_{2.5}$ para esos días y en naranja la predicción del modelo. Observamos que, cuando los valores reales son muy altos, la predicción está muy alejada del valor real que ocurrió. Por eso, aunque las tasas de error suban ligeramente, vamos a aumentar el parámetro de Minkowski a 3, con lo que damos más peso a estos eventos en el cálculo del error total y, aunque no mejoramos directamente la predicción de las singularidades, les damos más importancia en el conjunto global (de ahí la elección de este tipo de error). Hay que resaltar que Minkowski no puede usarse si el algoritmo de entrenamiento es Levenberg-Marquardt, por lo que se usa el método Quasi-Newton.

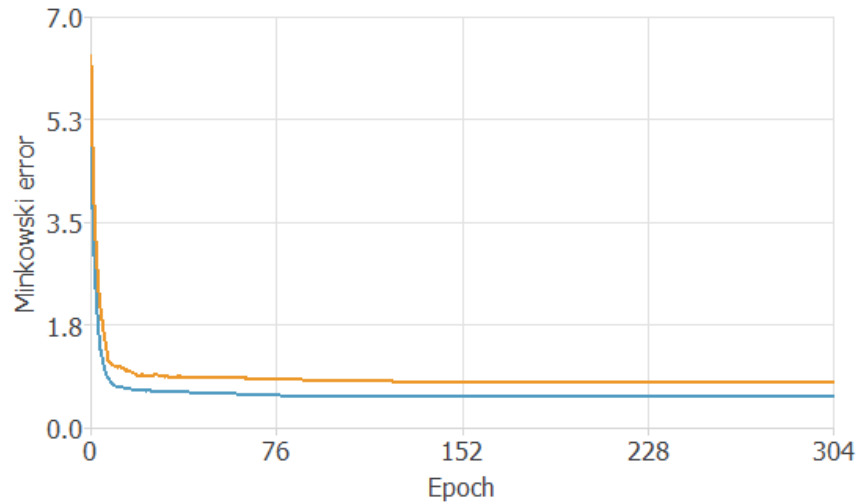


Figura X. Descenso del error de Minkowski en el entrenamiento.

En la figura X vemos el descenso del error a lo largo del entrenamiento con Quasi-Newton. Vemos que desciende rápidamente en los primeros epoch y van convergiendo el error de selección (línea naranja) y el de entrenamiento (línea azul).

Además, vamos a introducir un parámetro de regularización a la función de coste para controlar la complejidad de la red neuronal, sobre todo debida a la existencia de siete steps ahead. Las fórmulas de regularización son las siguientes:

$$l1_regularization = regularization_weight \cdot \sum |parameters|$$

$$l2_regularization = regularization_weight \cdot \sum parameters^2$$

La variación en los valores obtenidos se muestra en la tabla X.

	Ninguna	L1	L2
PM_{2.5}	5,54%	4,93%	5,52%
PM₁₀	2,05%	2,15%	2,11%
O₃	2,00%	2,06%	2,08%
NO₂	7,87%	8,42%	8,23%
SO₂	4,03%	3,54%	5,22%

Tabla X. Comparación de errores entre los parámetros de regularización.

A partir de estos datos, se decide un método de regularización L1, que usa la suma del valor absoluto de los parámetros. Esto se debe a la mejora en la predicción de $PM_{2.5}$, que es el contaminante más peligroso para la salud humana, aunque se pierde eficacia en la predicción del NO_2 .

5. 2. 5. Validación de los resultados

La relación más importante en la validación de los resultados es la existente entre los valores reales pasados y los valores que predijo nuestro modelo para esas fechas. Por eso, en las siguientes cinco figuras se muestra esta relación para cada uno de los contaminantes medidos a lo largo del último, siendo la línea azul los valores reales y la línea naranja la predicción de nuestro modelo.

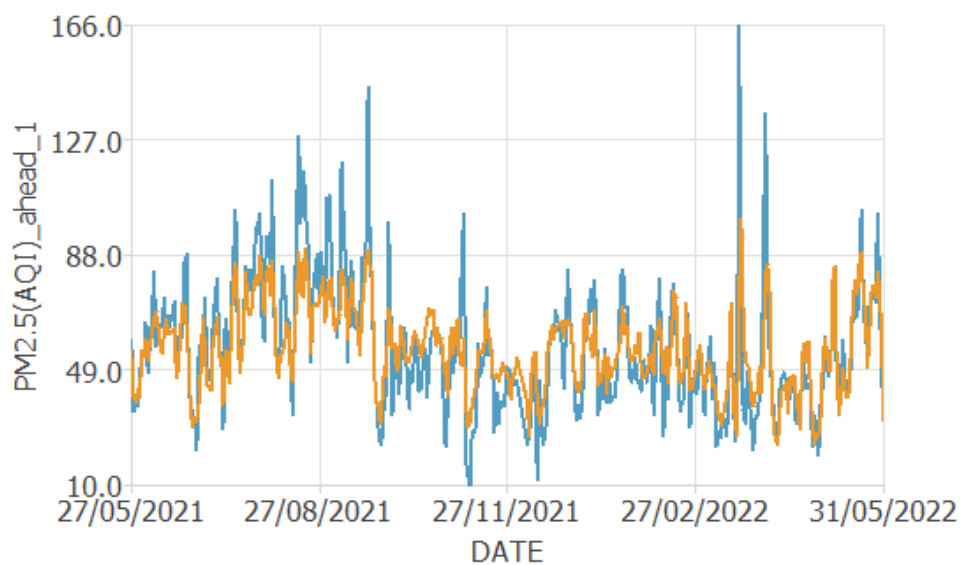


Figura X. Predicción de los valores de $PM_{2.5}$.

Para el $PM_{2.5}$ observamos que, aunque la predicción es bastante buena, los picos son el gran problema, ya que nos alejamos en gran medida del valor real que tuvo lugar ese día.

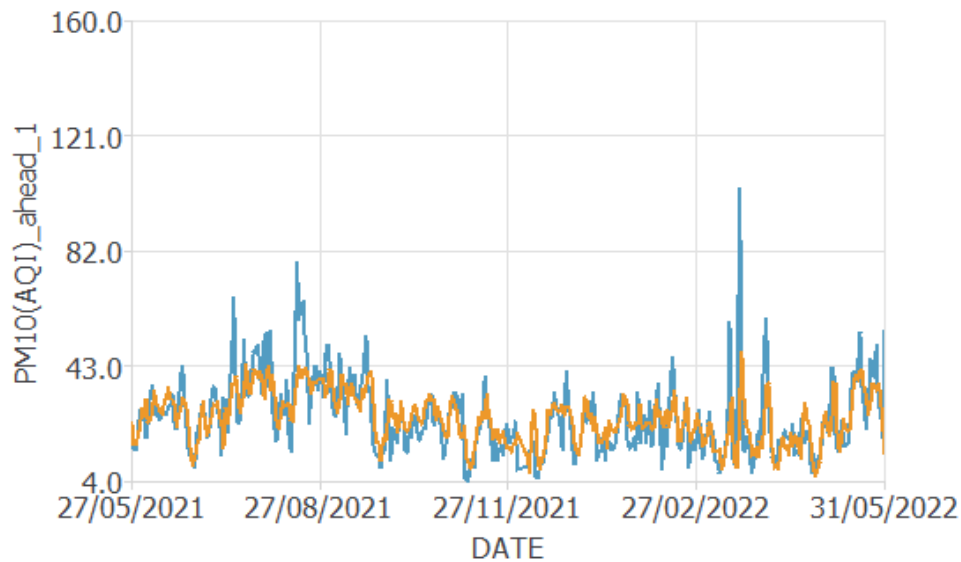


Figura X. Predicción de los valores de PM_{10} .

Algo similar ocurre para el PM_{10} , donde la predicción es muy buena en los valores medios, pero no se adapta demasiado bien a los eventos singulares.

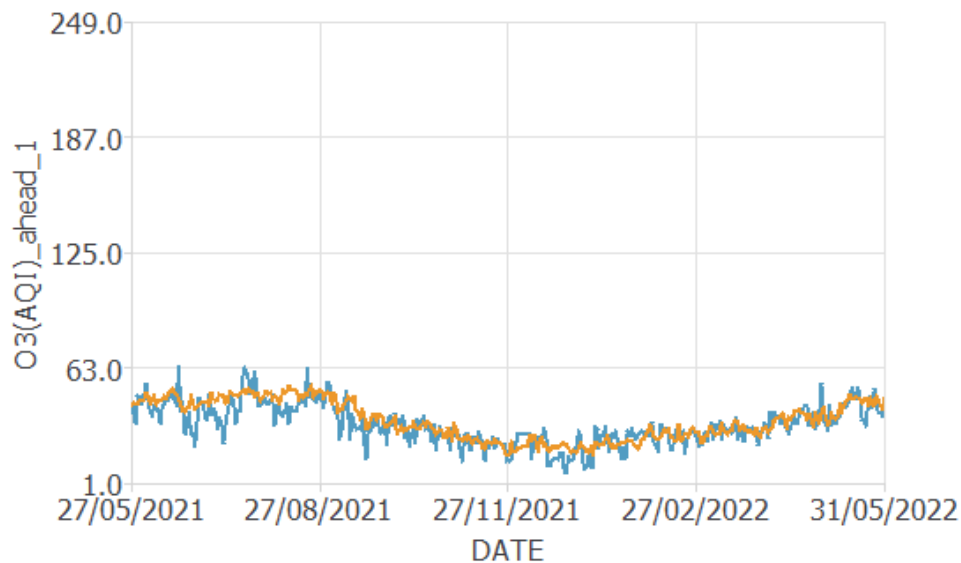


Figura X. Predicción de los valores de O_3 .

Para el ozono conseguimos una predicción muy buena, ya que como este contaminante no suele tener picos sino que es relativamente constante a lo largo del año, el error es prácticamente mínimo.

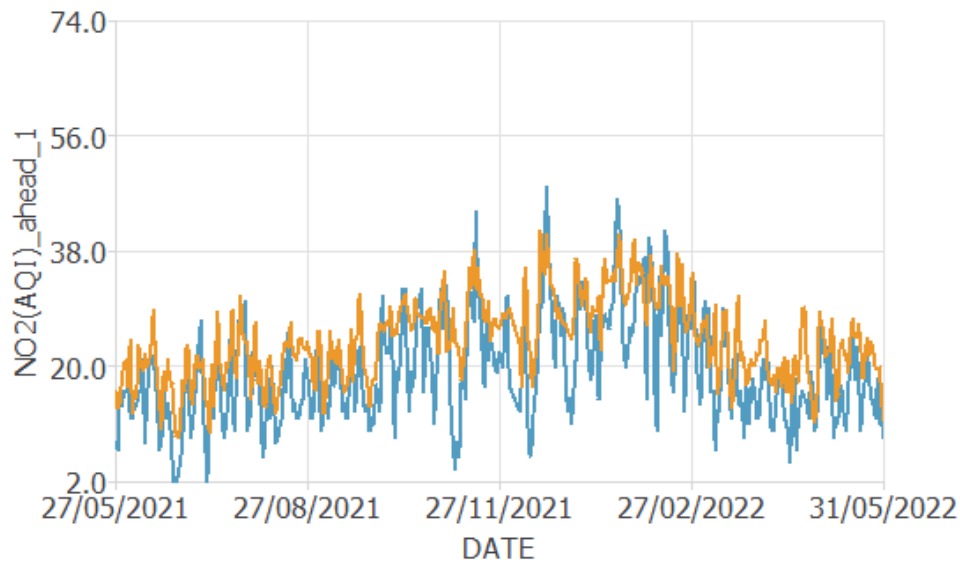


Figura X. Predicción de los valores de NO2.

En cuanto al dióxido de nitrógeno, la predicción que obtenemos suele tener niveles bajos de error, siguiendo la línea de tendencia que marcan los valores reales. Además, en este caso sí que se ajusta bien a los picos existentes. Sin embargo, casi siempre los valores reales son más bajos que los predichos (puede deberse a que las muestras históricas eran más altas que las actuales, ya que en los últimos años se han conseguido reducir los valores de este contaminante), por lo que aumenta ligeramente el error.

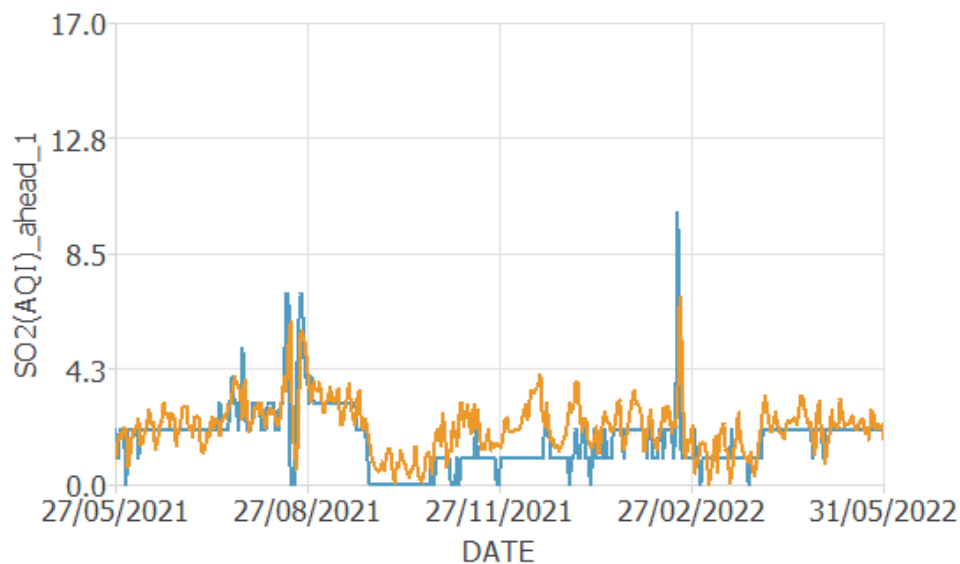


Figura X. Predicción de los valores de SO2.

Por último, la predicción del dióxido de azufre también es satisfactoria. Debido a los bajos niveles de este contaminante, siempre cercanos a cero, la predicción oscila más que los valores reales, pero como son tan bajos el error es mínimo. Como podemos

observar en el pico cercano al 27 de febrero de 2022, el modelo se adapta bien a los eventos singulares de este contaminante.

Seguidamente, se muestra la tabla X con los errores obtenidos para cada una de las variables a corto plazo (un step ahead) y a largo plazo (siete steps ahead). Vemos que hemos conseguido reducir el error para todas las variables respecto al comienzo del diseño del modelo, que se mostraba en la figura X.

Error final	
PM_{2.5} (1 step ahead)	4,44%
PM₁₀ (1 step ahead)	1,95%
O₃ (1 step ahead)	1,89%
NO₂ (1 step ahead)	8,32%
SO₂ (1 step ahead)	3,55%
PM_{2.5} (7 step ahead)	9,05%
PM₁₀ (7 step ahead)	2,56%
O₃ (7 step ahead)	2,26%
NO₂ (7 step ahead)	12,80%
SO₂ (7 step ahead)	7,05%

Tabla X. Resultados de error finales.

Observamos que los niveles de error son muy buenos para el PM₁₀, el O₃ y el SO₂. También son bastante buenos para el PM_{2.5}, y algo peores en el caso del NO₂, aunque siguen estando por debajo de un 15%, por lo que en general nuestro modelo predice de manera adecuada los valores de contaminación.

Para profundizar en estos errores y ver su distribución, se muestran las siguientes figuras. Las barras muestran el porcentaje de veces que el error ha estado entre los valores que indica el eje vertical. Las de gran tamaño quieren decir, por lo tanto, que se ha tenido un error en ese rango muchas veces, mientras que a medida que disminuyen significa que en esos rangos el error no ha estado prácticamente nunca.

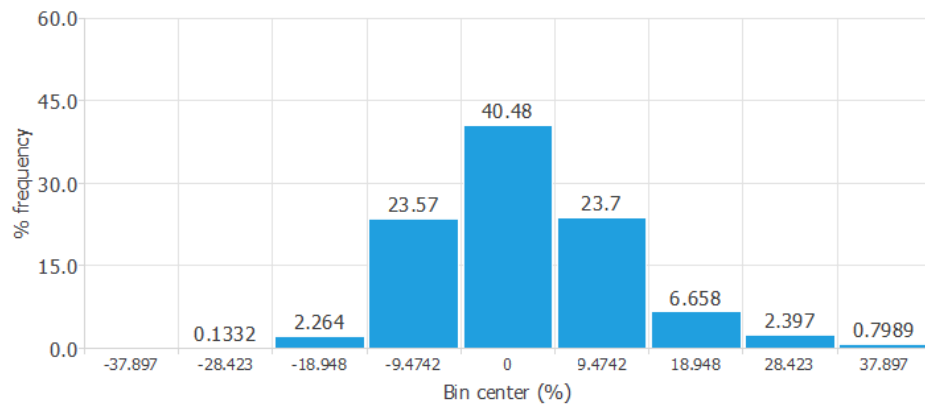


Figura X. Distribución del error para el PM_{2.5}.

En la figura X vemos la del PM_{2.5}. Un 40% de las veces está en el 0 y hay muy poca dispersión, por lo que los resultados son muy buenos.

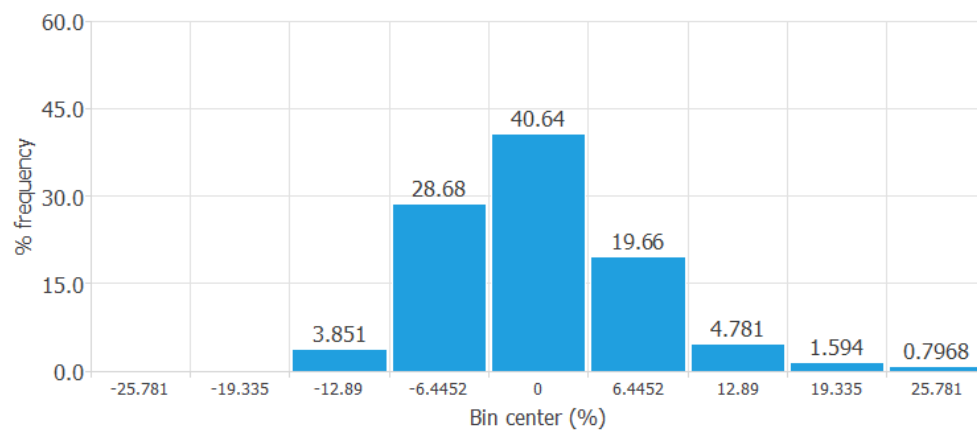


Figura X. Distribución del error para el PM₁₀.

En la figura X vemos la del PM₁₀. Los números son muy similares que para el PM_{2.5}, por lo que los resultados también son muy buenos en este caso.

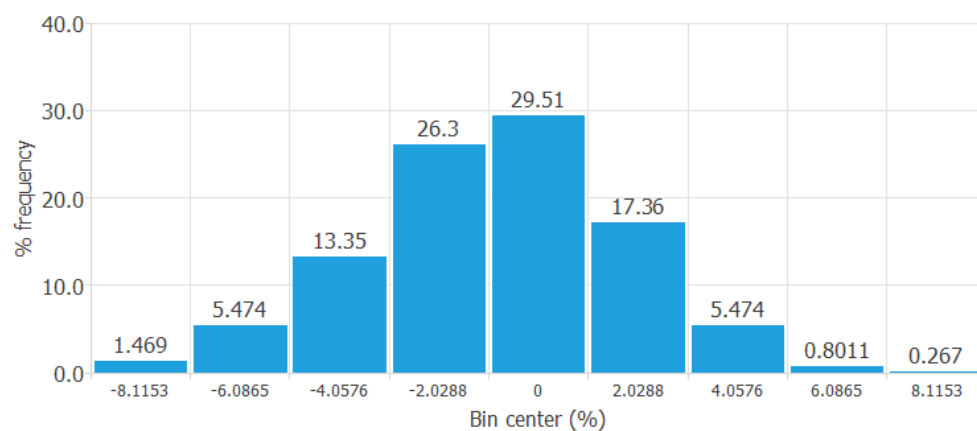


Figura X. Distribución del error para el O₃.

En la figura X vemos la del O3. Los resultados son todavía mejores que para la materia particulada, ya que, aunque hay menos porcentaje de error nulo, alrededor del 73% de las veces hay un error menor al 2%, lo que es un resultado tremendamente fiable.

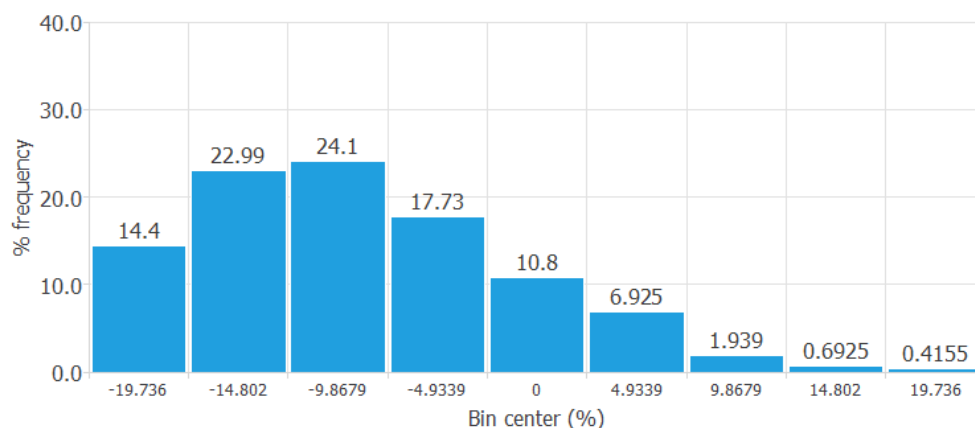


Figura X. Distribución del error para el NO2.

En la figura X vemos la del NO2. Como se puede observar, es el contaminante para el que se obtiene la peor predicción, con bastante diferencia respecto a los demás. Aun así, solo en un 15% de los casos el error es mayor del 15%, por lo que no son malos, aunque no lleguen a la gran precisión del resto. También ratificamos conclusiones obtenidas anteriormente, donde veíamos que la predicción del NO2 siempre es mayor que los valores reales (de ahí el error negativo).

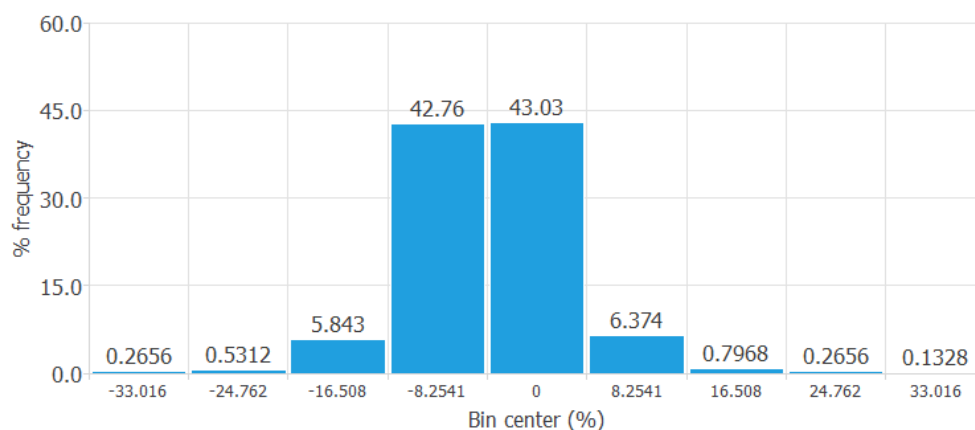


Figura X. Distribución del error para el SO2.

Por último, en la figura X vemos la del SO2. Los resultados son muy buenos, al tener prácticamente la mitad de las veces un error nulo y el resto un error pequeño menor del 10%.

5. 2. 6. Exportación del modelo

Podemos exportar el modelo para poder visualizar la expresión matemática resultante y utilizarlo donde sea necesario. Debido a la gran cantidad de variables, pesos y sesgos existentes en el mismo, ocuparía un número muy grande de páginas en este documento, por lo que se muestran solo algunas líneas de ejemplo de cada capa.

```
scaled_PM2.5(AQI)_lag_0 = (PM2.5(AQI)_lag_0-54.4776001)/19.5258007;
scaled_PM10(AQI)_lag_0 = (PM10(AQI)_lag_0-24.53549957)/11.86229992;
scaled_O3(AQI)_lag_0 = (O3(AQI)_lag_0-32.79840088)/14.54819965;
scaled_NO2(AQI)_lag_0 = (NO2(AQI)_lag_0-24.17060089)/10.29259968;
scaled_SO2(AQI)_lag_0 = (SO2(AQI)_lag_0-3.113509893)/2.05302;
...

perceptron_layer_1_output_0 = tanh( -2.02123
+(scaled_PRECIPITATION(mm)_lag_1 *0.0338929) + (scaled_TAVG(C)_lag_1*-
0.411994) +(scaled_TMAX(C)_lag_1 *0.391274) +
(scaled_TMIN(C)_lag_1*0.728711) + (scaled_PRESSURE(hPa)_lag_1 *0.392377) +
(scaled_WINDSPEED(km/h)_lag_1*-0.463761) +
(scaled_HUMIDITY(percentage)_lag_1*-0.440257) + (scaled_DAY_lag_0*-0.310097)
+ (scaled_MONTH_lag_0*0.335254) + (scaled_WEEKDAY_lag_0*0.062457) +
(scaled_PM2.5(AQI)_lag_0*0.299843) +...

perceptron_layer_2_output_0 = ( 0.96769 +
(perceptron_layer_1_output_0*0.279869) + (perceptron_layer_1_output_1*-
0.412671)+(perceptron_layer_1_output_2*0.232407)
+(perceptron_layer_1_output_3*0.0977992)+(perceptron_layer_1_output_4*0.116731)
+(perceptron_layer_1_output_5*0.213226)+(perceptron_layer_1_output_6*-
0.151479)+(perceptron_layer_1_output_7*-0.0232102)+(perceptron_layer_1_output_8
*1.12721) + (perceptron_layer_1_output_9*-0.163324) );
...

unscaling_layer_output_0 = 10+0.5*(perceptron_layer_2_output_0+1)*(166-10);
unscaling_layer_output_1 = 4+0.5*(perceptron_layer_2_output_1+1)*(160-4);
unscaling_layer_output_2 = 1+0.5*(perceptron_layer_2_output_2+1)*(249-1);
unscaling_layer_output_3 = 1+0.5*(perceptron_layer_2_output_3+1)*(74-1);
unscaling_layer_output_4 = 0+0.5*(perceptron_layer_2_output_4+1)*(17-0);
unscaling_layer_output_5 = 10+0.5*(perceptron_layer_2_output_5+1)*(166-10);
unscaling_layer_output_6 = 4+0.5*(perceptron_layer_2_output_6+1)*(160-4);
unscaling_layer_output_7 = 1+0.5*(perceptron_layer_2_output_7+1)*(249-1);
unscaling_layer_output_8 = 1+0.5*(perceptron_layer_2_output_8+1)*(74-1);
unscaling_layer_output_9 = 0+0.5*(perceptron_layer_2_output_9+1)*(17-0);
unscaling_layer_output_10 = 10+0.5*(perceptron_layer_2_output_10+1)*(166-
10);
unscaling_layer_output_11 = 4+0.5*(perceptron_layer_2_output_11+1)*(160-4);
...
```


5. 3. Interfaz web para la visualización de resultados

Un objetivo del trabajo es acercar el modelo creado a cualquier posible usuario. Esto permitiría que, por ejemplo, médicos sin conocimientos de inteligencia artificial puedan variar las recomendaciones a sus pacientes con problemas respiratorios tras visualizar la predicción conseguida anteriormente.

Por ello, es importante abstraer al usuario final del proceso de obtención de los datos, el procesado y el cálculo de la predicción, centrándonos solamente en el resultado final y el significado de este.

Para esta misión, se va a diseñar una interfaz web básica donde, a través de gráficas intuitivas en lugar de mucha cantidad de texto, una persona pueda conocer cómo va a ser la calidad del aire la próxima semana rápidamente.

Como se ha mencionado en el capítulo X, se utiliza Node y Express. Node permite crear herramientas de lado servidor de manera sencilla usando JavaScript, ofreciendo gran cantidad de paquetes que nos serán útiles en la creación de la interfaz. Express es un framework que ofrece mecanismos para renderización de vistas, aportando la parte gráfica.

Los lenguajes de programación que utilizaremos son el trío de HTML, CSS y JavaScript. Para hacer más rápida la codificación de HTML y evitar etiquetas y demás elementos, se usa Pug como preprocesador de HTML. La sintaxis es más simple lo que permite ahorrar tiempo. Por ejemplo, en la figura X se muestra la diferencia a la hora de escribir un “Hola Mundo”.



Figura X. Ejemplo diferencia HTML/Pug.

5. 3. 1. Diseño

La página contiene un encabezado con el título, una parte central con el contenido que se divide en datos actuales, gráficas de predicción y breve explicación de los datos mostrados, con enlaces a las fuentes, y un pie de página con información del trabajo. En la figura X se muestra un recorte con algunos de estos elementos.

Para crear las gráficas con la predicción de cada uno de los contaminantes, se utiliza la librería Charts.js. Permite crear el tipo de gráfica (barras, lineales...) que se desee simplemente introduciendo los datos y las opciones (tamaño, fuente de los ejes, título, etc.). Los colores de las columnas cambian dependiendo del valor, siguiendo el AQI (verdes si la predicción es menor que 50, amarillas entre 50 y 100 y sucesivamente).

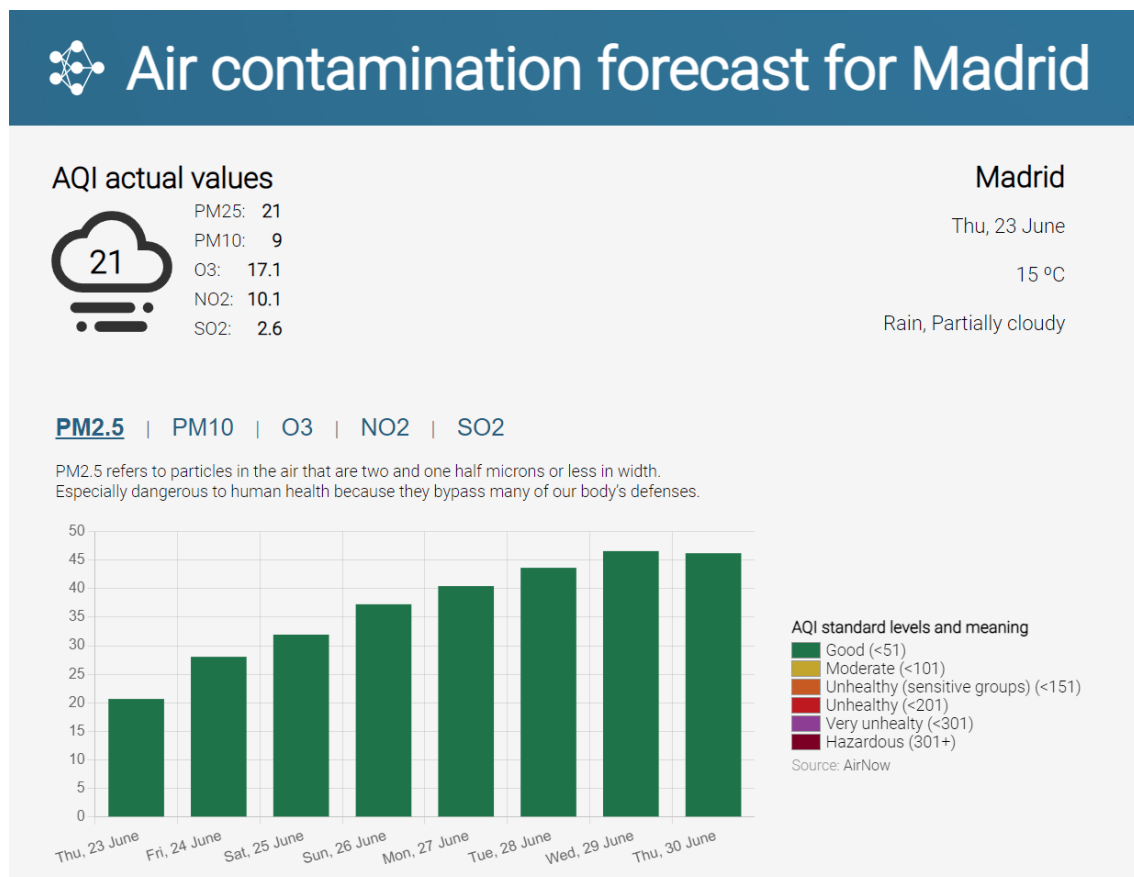


Figura X. Captura de pantalla de la interfaz web.

Sin embargo, para la hoja de estilos no hemos usado preprocesadores como puede ser Less, ya que debido a la sencillez de nuestro fichero no es de gran utilidad y no ahorraríamos demasiados recursos.

Respecto a la selección de colores y el diseño, se ha optado por un enfoque básico. Debido a que en las gráficas pueden aparecer hasta seis colores, utilizar muchos más en el resto de la página da una sensación de sobrecarga. Por lo tanto, se utiliza un azul con varios tonos, pero similares en el encabezado y pie de página, además de un fondo blanco para el contenido (resaltando la parte importante, que son las gráficas), con texto negro. El texto del encabezado y pie de página es blanco para poder leerse fácilmente en fondo azul, además de ser igual que el fondo del contenido, creando un patrón.

Se busca una sensación de seriedad y profesionalismo propia de páginas relacionadas con el mundo científico como es esta. Centramos la atención del visitante en la parte importante, siendo el resto añadidos en caso de que se busque información extra.

5. 3. 2. Funcionalidad

En cuanto a los scripts de la interfaz web, existen dos ficheros JavaScript que aportan la funcionalidad a la página. El fichero principal (index.js) contiene la mayoría de la lógica. Su misión principal es cargar el framework con las vistas, pero tiene más tareas.

Además, aquí se realizan las llamadas a las API para obtener los datos necesarios posteriormente en el modelo. Utilizamos dos API: la primera es <https://www.visualcrossing.com/weather-api> , para recoger la información meteorológica. Como necesitamos tanto la actual como la pasada, son necesarias dos llamadas. La segunda API es <https://aqicn.org/api/es/> , para recoger la información actual de los contaminantes en Madrid. En ambos casos obtenemos los datos en formato JSON, como se ve en la figura X, facilitando el uso posterior de los mismos.

```
queryCost: 1
latitude: 40.4196
longitude: -3.69196
resolvedAddress: "Madrid, Comunidad de Madrid, España"
address: "madrid"
timezone: "Europe/Madrid"
tzoffset: 2
▼ description: "Similar temperatures continuing with no rain expected."
▼ days:
  ▼ 0:
    datetime: "2022-06-23"
    datetimeEpoch: 1655935200
    tempmax: 23.7
    tempmin: 15.1
    temp: 19.4
    feelslikemax: 23.7
    feelslikemin: 15.1
    feelslike: 19.4
    dew: 9.2
    humidity: 54.2
    precip: 0.2
    precipprob: 100
    precipcover: 8.33
```

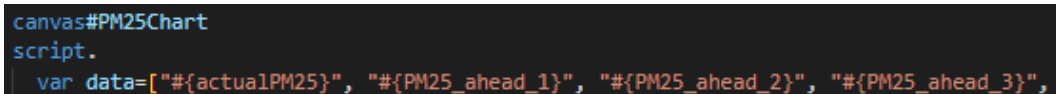
Figura X. Extracto del JSON con la información meteorológica del día actual

Las llamadas se utilizan utilizando la librería Axios, que es un cliente HTTP basado en promesas. De esta manera, proporcionamos los parámetros adecuados al método GET de la librería y nos quedamos esperando a recibir el objeto desde la API.

Debido a que el conjunto de datos utiliza referencias temporales, existe un método para obtener las fechas de la semana siguiente, permitiendo usarlas en el modelo y como eje horizontal de las gráficas mostradas en la vista.

Por otro lado, en este script se encuentra el código del modelo creado por la red neuronal en el capítulo anterior. Para optimizarlo y no tener un método megalítico con gran cantidad de variables, se ha dividido en pequeños métodos (uno por cada capa de la red) que reciben un vector de entrada y retornan un vector de salida tras realizar los cálculos oportunos.

Los valores usados en la vista se le pasan a través del método render de Express, que carga el framework. Al utilizar Pug, se pueden utilizar como variables normales de cualquier otro lenguaje de programación, lo que es una de las grandes ventajas respecto a HTML plano.

A screenshot of a code editor showing JavaScript code. The first line is `canvas#PM25Chart` in blue. The second line is `script.` in blue. The third line is `var data=["#{actualPM25}", "#{PM25_ahead_1}", "#{PM25_ahead_2}", "#{PM25_ahead_3}",` in red. The code is displayed on a dark background with syntax highlighting.

```
canvas#PM25Chart
script.
var data=["#{actualPM25}", "#{PM25_ahead_1}", "#{PM25_ahead_2}", "#{PM25_ahead_3}",
```

Figura X. Uso de datos obtenidos por el modelo en la creación de gráficas.

Por último, existe un segundo fichero JavaScript, con la diferencia de que este se carga una vez mostrada la vista, a diferencia del primero. Solamente contiene la funcionalidad de los botones que permiten la elección del contaminante que se muestra en las gráficas.

6. Conclusiones

La predicción de la contaminación atmosférica, y de cualquier valor dependiente de información meteorológica, es un problema complejo debido a la gran cantidad de factores que influyen en el proceso.

En este trabajo se ha conseguido realizar una predicción con un nivel de error muy bajo (siempre menor del 15%, llegando a niveles tan bajos como un 2% en algunos casos) para cada uno de los cinco contaminantes, cumpliendo el objetivo principal marcado antes de la realización de este. En cuanto a la recogida de datos históricos, se han obtenido todas las variables necesitadas, tanto de contaminación como meteorológicas, al igual que para los datos en tiempo real.

Para la consecución de este objetivo, ha sido necesaria la inmersión en el mundo de la inteligencia artificial y las redes neuronales, obteniendo información, seleccionando y aplicando los conceptos necesarios en el proyecto.

Por otro lado, se ha creado una interfaz web clara y concisa donde se ha integrado el modelo. Esta permite la visualización de resultados con un despliegue continuo, calculando las variables objetivo para la próxima semana, en lugar de para una semana en concreto. La abstracción para el usuario es completa, por lo que se cumple el objetivo de acercar el sistema a usuarios inexpertos, permitiendo así a estos el análisis de los resultados obtenidos.

Finalmente, el uso de Design Science Research Methodology me ha hecho conocer una metodología diferente a las vistas a lo largo del grado, y sus variaciones respecto a las tradicionales al estar enfocada al mundo científico. Por otra parte, la utilización de Scrum en un trabajo real me ha servido para aplicar las ventajas de la metodología ágil que conocía de manera teórica, y por lo tanto profundizar en su conocimiento.

En resumen, la realización de este trabajo me ha permitido explorar una rama de la informática sobre la que no conocía gran cosa aparte de lo visto en algunas asignaturas, y tras estos meses he aprendido bastante sobre conceptos teóricos además de aplicarlos correctamente sobre un problema real. También me ha dado la oportunidad de utilizar conocimientos aprendidos a lo largo del grado como el uso de una metodología de desarrollo de software.

6. 1. Líneas futuras

Como ampliaciones a este trabajo se plantean una serie de opciones de mejora.

En primer lugar, implementar técnicas de “continuous training”, basadas en el reentrenamiento del modelo conseguido con los nuevos datos que se van obteniendo día a día, dando más importancia a las tendencias actuales. De esta manera, podría reducirse aún más el error en casos como el del dióxido de nitrógeno, donde los valores han cambiado bastante a lo largo de los últimos años.

Por otro lado, se podrían añadir capas neuronales de otros tipos que podrían funcionar bien para este proyecto, como pueden ser las convolucionales (usadas sobre todo en el análisis automático de imágenes).

Respecto a la interfaz web, se podría añadir funcionalidad adicional como podría ser la descarga de los datos pasados por parte del usuario o la exportación de un fichero con las predicciones para su visualización fuera de las gráficas.

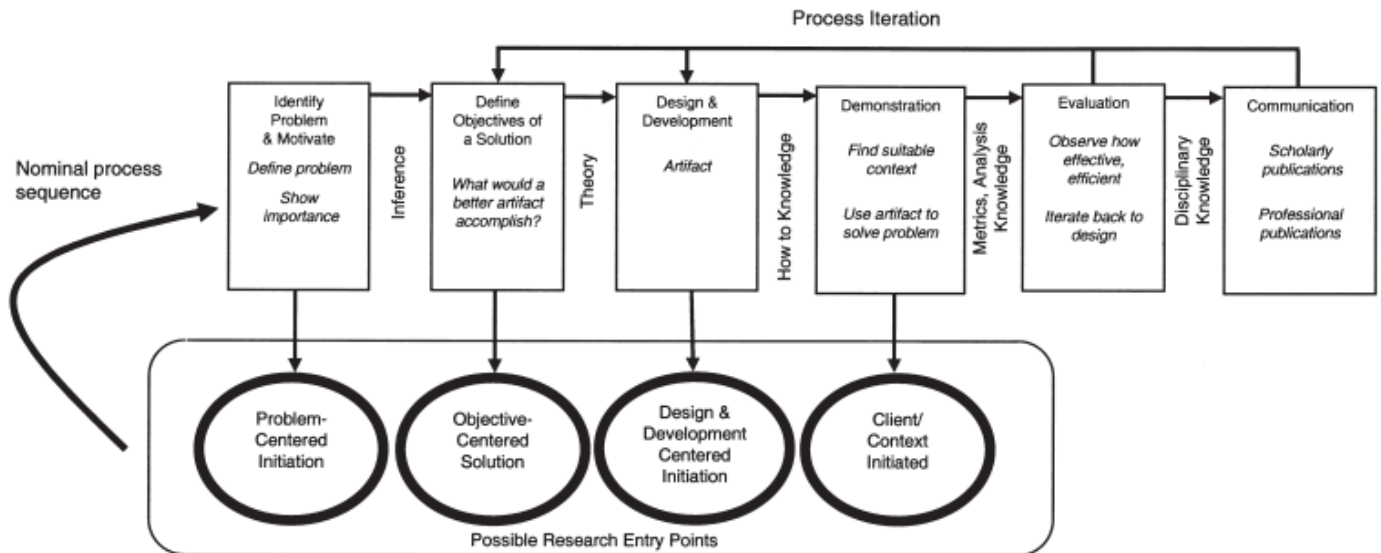
Referencias

[1] b

[2] b

Anexo técnico

Anexo I. Plan del proyecto software



Identificación del problema y motivación: en las últimas décadas ha habido un gran incremento en los niveles de contaminación en las grandes ciudades, por lo tanto, se necesitan métodos que permitan la predicción de estos valores de manera precisa y rápida para poder tomar medidas a tiempo. La motivación es salvaguardar la salud pública y preservar el medio ambiente.

Objetivos de la solución: obtener una predicción de los valores para cinco contaminantes en la ciudad de Madrid, con el error mínimo posible. Además, dar una forma de visualización sencilla e intuitiva de estos datos al pública general.

Diseño y desarrollo: se ha creado el archivo modelo.c que se puede integrar en cualquier sistema, a través del desarrollo de una red neuronal que utiliza datos históricos para obtener la predicción de los valores futuros.

Demonstración: uso del modelo previamente creado para obtener unos valores en la ciudad de Madrid, obteniendo 35 salidas (una por día de la semana para cada contaminante).

Evaluación: gráficas de validación, XX memoria. Se compara la serie temporal de un contaminante en concreto con la predicción obtenida por el modelo, obteniendo la tasa de error.

Comunicación: interfaz web con gráficos de barras que muestran la evolución para la próxima semana de los niveles de contaminación atmosférica, además de una breve explicación de la leyenda y método utilizados.

A continuación, se muestra el **Product Backlog** inicial para este trabajo, que es un artefacto de la metodología Scrum, que es una lista con prioridades de ítems que deben ser completados como parte de un proyecto mayor. Estas tareas se dividen en los distintos sprint backlog.

Ítems	Estimación (días/4h)	Prioridad
Decidir los contaminantes atmosféricos a predecir	2	1
Conseguir datos históricos para esos contaminantes	3	2
Conseguir datos históricos de información meteorológica	3	3
Conseguir datos en tiempo real de contaminación	3	4
Conseguir datos meteorológicos en tiempo real	3	5
Diseño de la red neuronal y elección de parámetros	7	6
Pruebas y mejoras del modelo obtenido	14	7
Diseño del frontend	14	8
Integración del modelo en la interfaz web	2	9
Documentación y memoria	14	10
Total	65	

Sprints

Sprints	Fechas	Tareas
Sprint 1	1ª quincena marzo	1, 2, 3
Sprint 2	2ª quincena marzo	4, 5
Sprint 3	1ª quincena abril	6
Sprint 4	2ª quincena abril	7
Sprint 5	1ª quincena mayo	7
Sprint 6	2ª quincena mayo	8
Sprint 7	1ª quincena junio	8, 9, 10
Sprint 8	2ª quincena junio	10

<https://edu.gcfglobal.org/es/scrum/como-crear-un-plan-de-trabajo-scrum/1/>

Anexo II. Especificación de requisitos del software

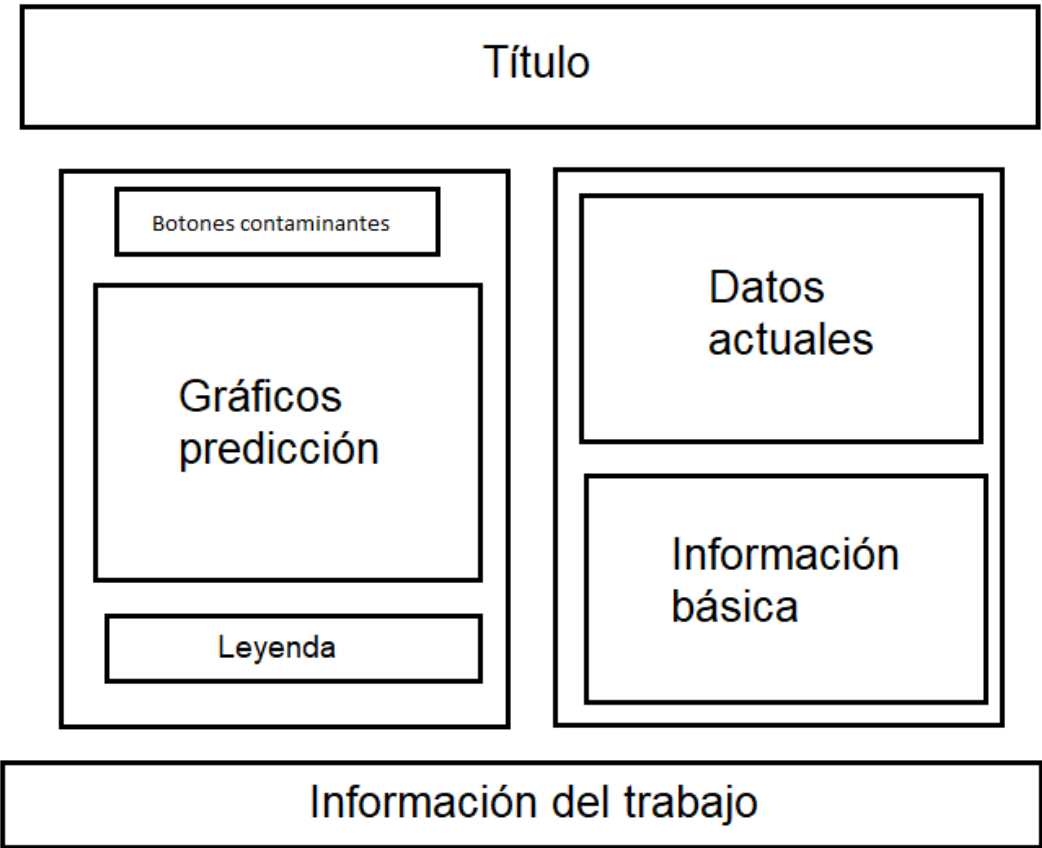
Historias de usuario:

- Quiero obtener un modelo que me permita predecir los valores de contaminación atmosférica en Madrid para la próxima semana, para poder tomar las medidas acordes.
- Quiero poder visualizar la predicción de manera sencilla e intuitiva, para entender los datos rápidamente.

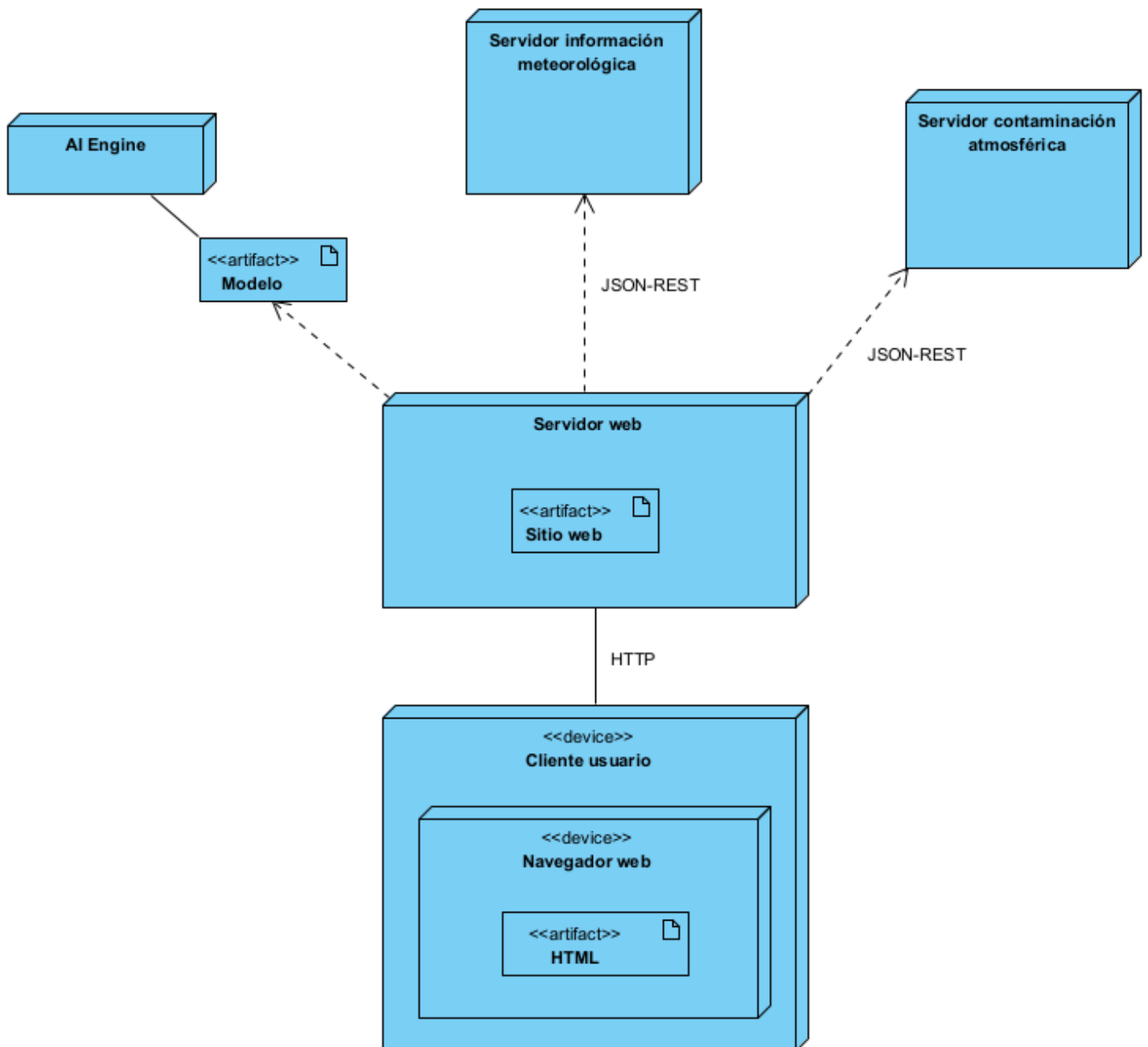
Anexo III. Especificación de diseño

Diseño de la interfaz de usuario

Prototipo



Despliegue e implementación



Anexo IV. Documentación técnica de programación

model.c (Modelo obtenido)

index.pug (Framework HTML)

style.css (Hoja de estilos)

index.js (JavaScript lado servidor)

main.js (JavaScript lado cliente)

Documentación de bibliotecas, código fuente, manual del programador. Info necesaria para modificar y mantener el sistema. (¿meter cosas de la web?)

Anexo V. Manuales de usuario

¿Visualización de las predicciones: acceder a la interfaz web, pulsar sobre el botón del contaminante deseado para ver la predicción.

RESUMEN y ABSTRACT

INTRODUCCION

REDES N

MODELO

IA

PARA QUE: PEREDECIR CONT. TIMEPO ATMOSFERICO

OBJETIVOS

CONCEPTOS TEORICOS

TFM PABLO PUEIDE ESTAR MUY BIEN

DIFERENCIA MODELO FISICO, IA

TECNICAS Y HERRAMENTAS

METODOLOGIA DESING SCIENCE, SCRUM

LENGUAJES PROGRAMACION

MOTOR

ASPECTOS RELEVANTES (PARTE MAS IMPORTANTE)

RECOGIDA DATOS HISTORICOS

DISEÑO MODELO

PRUEBA MODELO

MODELO FINAL: SEGUIR APARTADOS PABLO

COMPARACION FISICOS

FRONTEND

CONCLUSION

REFERENCIAS

ANEXO TECNICO

PLAN PROYECTO

REQUISITOS

DISEÑO

APLICACION WEB

DISEÑO MODELO

...

CODIGO

MANUAL DE USUARIO