UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

Análisis de datos de temperatura y humedad de suelo procedentes de sensores IoT desplegados en un viñedo

Presentado por Gabriel Hernández Vallejo
en Universidad de Burgos — 3 de junio
de 2023
Tutores: Rubén Ruiz González, Alejandro
Merino Gómez

UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

D. Rubén Ruiz González y D. Alejandro Merino Gómez, profesores del departamento de Ingeniería Electromecánica, área de Ingeniería de Sistemas y Automática.

Exponen:

Que el alumno D. Gabriel Hernández Vallejo, con DNI 71709111-X, ha realizado el Trabajo final de Grado en Ingeniería Informática titulado *"Análisis de datos de temperatura y humedad de suelo procedentes de sensores IoT desplegados en un viñedo"*.

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

En Burgos, 3 de junio de 2023

Vº. Bº. del Tutor:	Vº. Bº. del co-tutor:
D. nombre tutor	D. nombre co-tutor

Resumen

En este primer apartado se hace una **breve** presentación del tema que se aborda en el proyecto.

Descriptores

Palabras separadas por comas que identifiquen el contenido del proyecto Ej: servidor web, buscador de vuelos, android ...

Abstract

A **brief** presentation of the topic addressed in the project.

Keywords

keywords separated by commas.

Índice general

Índice de figuras

Índice de tablas

Introducción

El cuidado de las tierras de cultivo vitícolas ha supuesto a menudo a lo largo de la historia diferentes retos desde el comienzo de su domesticación entorno al 3500-3100 a.C., continuando en la época del Antiguo Egipto, lugar en el que se cree se comenzó a popularizar la ingesta del producto resultante de la labranza y cosecha de los cultivos: el vino [3].

En la península ibérica el interés por este proceso y su resultante comenzó en el periodo fenicio cerca del 1100 a.C., expandiéndose por todo el territorio peninsular gracias al comercio, lo que dio como resultado que los procesos enológicos fueran ya bien conocidos en los siglos IV-III a.C. [3, 4].

Entre los principales retos a los que se han enfrentado los viticultores en la península ibérica se encuentran numerosas plagas de insectos, ácaros, nematodos, diferentes vertebrados, moluscos, bacterias, etc [5]. Siendo la más devastadora la causada por la histórica palga de filoxera (*Phylloxera vastatrix*) a finales del siglo XIX y comienzos del XX, que obligó a una reestructuración prácticamente completa de la viticultura española [6].

Sin embargo, el principal factor influyente en el cultivo de la vid es el clima, siendo este el causante de sus alteraciones, fisiopatías y la presencia de casi todas las plagas, de manera que estas se encuentran relacionadas con diferentes factores climáticos como la temperatura, las precipitaciones, la humedad y la humectación. De esta forma, por ejemplo, las altas lluvias o humedades y/o bajas temperaturas pueden producir corrimiento del racimo (escasez anormal de bayas en los racimos) o las altas temperaturas pueden generar desequilibrios hídricos que afecten a los injertos y a aquellas cepas infectadas por hongos de la madera [5].

Por otro lado, los datos climatológicos históricos indican que la temperatura global ha aumentado 1,1ºC desde el inicio de la Revolución Industrial hasta nuestros días, siendo este desequilibrio causado en los úiltimos años por la actividad humana, produciéndose a valocidad tal que los ecosistemas no tienen tiempo de adaptarse [7]. Las precipitaciones, además, no son uniformes sino que aumentan en zonas donde ya son muy abundantes y disminuyen en las regiones más secas [7].

Ante esta situación se plantea la cuestión de cuáles de estos efectos podríamos ser capaces de predecir y en qué proporción para, si bien no subsanar completamente, si ser capaces de paliar en cierta medida conociendo con anterioridad ciertos datos climatológicos que ayudarán a las diferentes tomas de decisiones de los viticultores.

De esta manera, siendo conocedores a corto o medio plazo la humedad y la temperatura superficial del suelo, entre otras variables, podríamos adaptar los sistemas de riego para adecuarlos a las necesidades de los cultivos y por otro lado, gestionar de forma adecuada los recursos hídricos nacionales (en este caso de la zona hidrográfica del Duero) al mismo tiempo que protegemos a las vides de plagas y trastornos fisiológicos debidos a diferentes factores como la humectación o la temperatura del suelo.

En este proyecto se propone el análisis de los datos de temperatura y humedad del suelo procedentes de sensores IoT (*Internet of Things*) desplegados en un viñedo situado en la zona sur de la provincia de Burgos, para, de esta forma, obtener modelos predictivos que nos permitan conocer el comportamiento de ciertas variables en función de otras y anticipar su evolución futura a corto plazo.

1.1. Estructura de la memoria

La memoria del proyecto cuenta con la siguiente estructura:

- Introducción: descripción del problema que el proyecto pretende resolver junto con la estructuración de la memoria y los materiales adjuntos.
- Objetivos del proyecto: descripción de los objetivos a cumplir con el desarrollo del proyecto.
- Conceptos teóricos: explicación de los conceptos teóricos necesarios para la comprensión del proyecto, el problema a abordar y la solución propuesta.

- Técnicas y herramientas: listado de técnicas y herramientas empleadas durante el desarrollo del proyecto.
- Aspectos relevantes del desarrollo: aspectos a destacar durante la realización del proyecto.
- Trabajos relacionados: trabajos relacionados con el problema que aborda el proyecto.
- Conclusiones y líneas de trabajo futuras: conclusiones obtenidas de la realización del proyecto y posibilidades de ampliarlo o de introducir mejoras.

Además de la memoria, se incluyen una serie de anexos:

- Plan de proyecto software: planificación temporal y estudio de viabilidad del proyecto.
- Comprensión del negocio: especificación de los objetivos y requsitos que el proyecto pretende abordar.
- Comprensión de los datos: estudio del conjunto de datos inicial y su adecación al problema.
- Preparación de los datos: proceso llevado a cabo para seleccionar los datos empleados en el modelado.
- Modelado: creación de los diferentes modelos.
- Evaluación: evaluación de los modelos obtenidos.
- Manual del programador: aspectos relevantes del código fuente del proyecto.
- Manual de usuario: guía de usuario para el manejo de la aplicación asociada al proyecto

1.2. Materiales adjuntos

El proyecto incluye el siguiente contenido:

- Python Notebook con el pre-procesamiento de datos y las diferentes expliciones.

- Python Notebook con las integración de los diferentes conjuntos de datos en un único fichero.
- Python Notebook con el modelado de la Red Neuronal Artificial empleada para la regresión de humedad y temperatura y las explicaciones correspondientes.
- Conjunto de datos de los sensores desplegados en el viñedo.
- Conjunto de datos del pluviómetro desplegado en el viñedo.
- Scripts de Python para mostrar las gráficas de los datos (de sensores y pluviómetro) sin procesar.
- Scripts de Python para mostrar las gráficas de los datos (de sensores y pluviómetro) procesados
- Script de Linux para la instalación del entorno virutal.
- Script de Windows PowerShell para la instalación del entorno virtual.
- Script de Windows CMD para la instalación del entorno virtual.

Los recursos mencionados se encuentran disponibles en GitHub [8].

Objetivos del proyecto

El proyecto aborda diferentes objetivos. Podemos hacer una distinción entre objetivos generales, objetivos técnicos y objetivos personales.

2.1. Objetivos generales

- Realizar un análisis de datos para ser capaces de predecir la temperatura y humedad del suelo en un viñedo.
- Facilitar la comprensión de los datos recogidos mediante representaciones gráficas.
- Buscar las correlaciones entre las diferentes medidas obtenidas y las diferentes variables ambientales y/o de características del suelo.
- Encontrar modelos predictivos que permitan explicar el comportamiento de ciertas variables en función de las otras y anticipar la evolución futura a corto plazo.

2.2. Objetivos técnicos

- Realizar visualizaciones de los datos recogidos por los sensores IoT (*Internet of Things*) desplegados en el viñedo.
- Realizar un pre-procesamiento de datos mediante librerías de manipulación de datos de Python como Pandas.

- Modelar una Red Neuronal Artificial con Keras que permita predecir la humedad y temperatura del suelo gracias a los datos proporcionados.
- Emplear Git como sistema de control de versiones distribuido mediante la plataforma GitHub.
- Emplear ZenHub para la gestión de proyectos mediante las metodologías ágiles.
- Aplicar en la medida de lo posible las metodologías ágiles mediante la técnica Scrum aprendida durante el desarrollo del grado (ciertos aspectos como las "*daily*" no pueden aplicarse debido al carácter del TFG).

2.3. Objetivos personales

- Emplear los conocimientos y técnicas adquiridas durante el desarrollo de los diferentes cursos del Grado en Ingeniería Informática.
- Profundizar en el uso de la inteligencia artificial para la resolución de problemas cotidianos que tendrán un reflejo en un sistema real (como es el campo del cultivo de la vid).
- Profundizar en la utilización de un lenguaje de programación tan versátil como es Python para el análisis de datos, empleando diferentes librerías nativas y de terceros, así como crear nuevos módulos.
- Profundizar en la librería de Tensorflow que se encuentra tan extendida en los diferentes ámbitos de la vida cotidiana en los que se emplean las redes neuronales (mundo laboral e investigador).

Conceptos teóricos

El desarrollo del proyecto cuenta con diferentes fases siguiendo el proceso de *descubrimiento de conocimiento en bases de datos, KDD*, compuesto por la comprensión del negocio, la comprensión de los datos, la preparación de datos, el modelado, la evaluación del modelo y el despliegue del producto software [1].

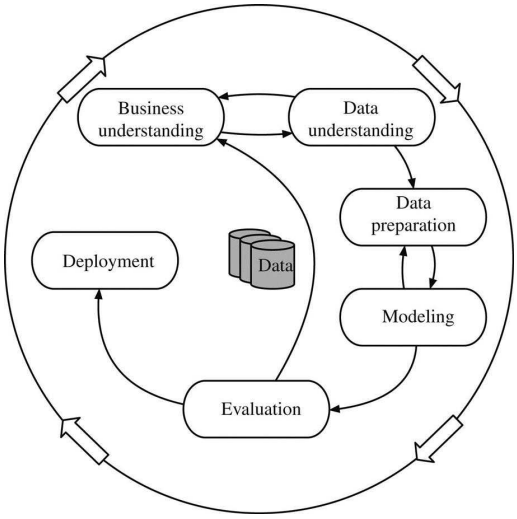

Figura 3.1: *Proceso KDD*. Extraido de [1]

En la fase de comprensión del negocio se analizan los objetivos y requisitos del proyecto. En este caso quedan bien marcados en la descripción del proyecto y la introducción, así como en la sección de los objetivos.

En la etapa de comprensión de los datos se crea el conjunto de datos inical y se comprueba si este es adecuado, de forma que si se determina que

no lo es se deberá continuar con la recopilación. En el caso que nos atañe como estamos sujetos a los plazos del curso la recopilación de más datos puede complicarse.

En la preparación de los datos se realiza el pre-procesamiento de estos, de forma que puedan ser empleados en el modelado.

En la fase de modelado, como su propio nombre indica, se crean los modelos, lo que se encuentra relacionado extrechamente con la fase de preparación, puesto que algunas herramientas de pre-procesamiento incluyen un modelo interno de los datos para transformarlos [1].

En la fase de la evaluación se estima el rendimiento del modelo, reconsiderándose en su caso los objetivos, de forma que si los modelos son poco efectivos se vuelve a la primera fase, ya que se trata de un proceso iterativo.

No se contempla realizar un despliegue de los modelos resultantes en ninguna aplicación, puesto que se sale del alcance del proyecto, sin embargo, se obtendrán resultados que pueden ser empleados en posteriores desarrollos de diferentes índoles.

En esta sección se presentarán los conceptos teóricos relevantes en cada etapa para facilitar su comprensión.

3.1. Pre-procesamiento de datos

En el pre-procesamiento de datos se pretende realizar la integración y limpieza de estos, de forma que disminuyan los posibles problemas de calidad que puedan surgir en los diferentes sistemas de información.

Como norma general el **proceso de integración** debe ser realizado durante la fase de recopilación de los datos. La **limpieza** permite la detección y corrección de los problemas no resueltos en la fase anterior como los valores anómalos (*Outliers*) o faltantes [9].

En este caso, los datos si bien no se encontrarán integrados de forma completa, estarán en formatos compatibles (tanto en lo referente a los nombres de los atributos como las dimensiones de los valores), de forma que no es necesario realizar un proceso exhaustivo, simplificando las operaciones a añadir exclusivamente una clase identificativa de los sensores concretos.

Tras la integración de las diferentes fuentes (e.g. bases de datos), se puede realizar un resumen de los atributos, en la que se mostrarán las características generales de estos como medias, mínimos, máximos y valores

posibles. En esta tabla podríamos obtener información trascendental para proceso de análisis, sobre todo para atributos categóricos. En el caso de atributos numéricos un mecanismo visual que es especialmente útil es la gráfica de dispersión [9], que es la técnica de visualización de datos que se empleará mayoritariamente durante el desarrollo del proyecto.

En el conjuto de datos podemos encontrar **valores faltantes**, que pueden ser reemplazados por diferentes razones. Una de ellas es que el modelo que empleemos puede no tratar bien estos valores o que utilice un mecanismo de tratamiento que no sea adecuado. Un problema asociado a su detección es que estos no estén representados como nulos, lo que puede introducir sesgo en el conocimiento extraido [9].

Ante esta situación se puede actuar de diferentes maneras [9]:

- Ignorarlos (ciertos algoritmos son tolerantes a los valores faltantes).
- Eliminar el atributo que contiene valores faltantes.
- Filtrar las filas: eliminar las filas afectadas, lo que introduce cierto sesgo en muchas ocasiones.
- Reemplazar el valor por otro que preserve la media y la varianza del conjunto de datos en caso de atributos numéricos y la moda en atributos catregóricos. Una forma de reemplazar los valores faltantes es la imputación de datos perdidos, que consiste en predecirlos a partir de otros ejemplos. Existen también algoritmos que se emplean tradicionalmente para este fin.
- Segmentar: se segmentan las filas por los valores disponibles y se obtiene un modelo por cada uno de los segmentos y se combinan.
- Modificar la política de calidad de datos y esperar a que los faltantes estén disponibles.

Además de las situaciones anteriores, es posible que el conjunto de datos cuente con **valores erróneos** que se deben detectar y tratar. La detección de estos campos puede realizarse de diferentes maneras.

En el caso que nos atañe se deben buscar los **valores extremos**, que no significa que sean erróneos, sino que estadísticamente se clasifican como anómalos, aunque representen un estado genuino de la realidad. Con todo y con eso, estos valores pueden suponer un problema para algunos métodos que se basan en el ajuste de pesos como las redes neuronales. En otras

ocasiones pueden haber datos erróneos que caen en la normalidad, por lo que no pueden ser detectados estadísticamente sin conocimiento explícito del dominio del problema [9].

No detectar estos valores puede resultar en problemas si posteriormente se normalizan los datos, puesto que la mayoria de datos estarían en un rango muy pequeño [9], mientras que unos pocos se encontrarán en rangos más grandes y alejados de estos, lo que puede introducir problemas de eficiencia y precisión.

El tratamiento de los datos erróneos o anómalos pueden ser tratados de forma similar a los faltantes [9]:

- Ignorarlos (ciertos algoritmos son robustos a datos anómalos).
- Eliminar el atributo que contiene los datos anómalos, por ejemplo si esta situación se produce continuamente (es preferible reemplazarla por otra columna con valores discretos estableciendo la corrección o no del valor).
- Filtrar las filas: eliminar las filas afectadas, lo que, de nuevo, puede introducir cierto sesgo.
- Reemplazar el valor por nulo, por los máximos o mínimos del atributo o por las medias (se debe tener en cuenta que los modelos sean capaces de procesar los valores nulos, ya que en caso contrario se deberá hacer frente a un nuevo problema en el conjunto de datos).
- Discretizar: transformar un valor continuo en uno discreto.

Los atributos con valores erróneos serán mas graves cuando este sea empleado como clase o valor de salida de la predicción [9], puesto que afectarán directamente al cálculo del rendimiento del modelo.

Filtro de Hodrick-Prescott

Además de los mecanismos descritos para realizar el pre-procesamiento de los datos, se emplea el filtro de Hodrick-Prescott, que permite extraer los componentes tendenciales y cíclicos de una secuencia temporal, de manera que de los originales se puedan utilizar las tendencias para entrenar los diferentes modelos [10].

Para el cáclculo del componente tendencial (τ_t), se emplea un valor positivo λ, de forma que se resuelve:

$$\text{mín} \sum_{t=1}^{T} (y_t - \tau_t)^2 + \lambda \sum_{t=2}^{T-1} [(\tau_{t+1} - \tau_t) - (\tau_t - \tau_{t-1})]^2 \tag{3.1}$$

Dónde y_t se corresponde con:

$$y_t = \tau_t + c_t \tag{3.2}$$

Siendo de esta forma la componente cíclica calculada cómo:

$$c_t = y_t - \tau_t \tag{3.3}$$

3.2. Modelado

Para la creación de los modelos emplearemos diferentes tipos de *Redes Neuronales Artificiales*, como el perceptrón multicapa (*MLP*), *Gated Recurrent Unit* y *Long Short-Term Memory*.

MLP

El perceptrón multicapa además de una capa de entrada y salida contiene varias capas intermedias llamadas ocultas (*hidden layers*), denominadas de este modo puesto que los cálculos internos están ocultos al usuario [2].

En este tipo de redes las señales se propagan inicialmente hacia delante para obtener un resultado, de forma que las neuronas de una capa se encuentran interconectadas con las de la capa siguiente y así sucesivamente en todas ellas (*feed-forward*).

Cada una de estas conexiones cuenta con un peso, que se inicializa generalmente de forma aleatoria. Estos son empleados para el cálculo de las señales de activación de cada neurona que se utilizarán en la función de activación:

$$\begin{aligned} \bar{h}_1 &= \Phi\left(W_1^T \bar{x}\right) \\ \bar{h}_{p+1} &= \Phi\left(W_{p+1}^T \bar{h}_p\right) \quad \forall p \in \{1 \dots k-1\} \\ \bar{o} &= \Phi\left(W_{k+1}^T \bar{h}_k\right) \end{aligned} \tag{3.4}$$

Donde $\bar{h}_1$ será el resultado de aplicar la función de activación en la primera capa oculta (la capa de entrada actua de intermedio entre las entradas y la *RNA*), $\bar{h}_{p+1}$ serán los resultados de la función de activación en la capa p-ésima y $\bar{o}$ en la capa de salida.

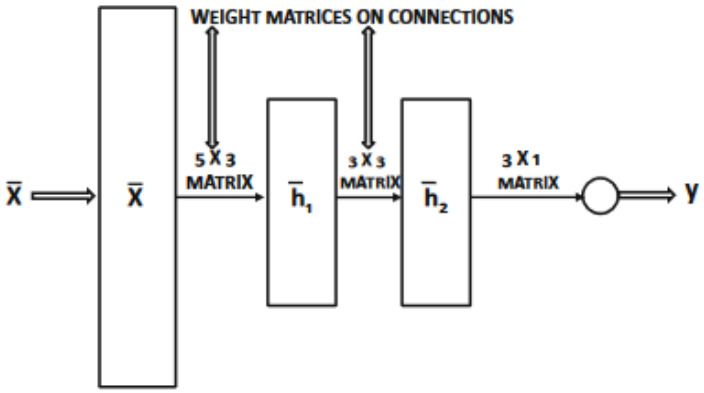

Figura 3.2: Perceptrón Multicapa. Extraido de [2]

Existen diversas funciones de activación entre las que destacan las sigmoides por ser aplicadas de forma elemental [2].

Tras la propagación de las señales de activación hacia adelante, se calculará el gradiente del error empleando el algoritmo de *Backpropagation*, que utilizará la regla de la cadena del cálculo diferencial para realizar este propósito, de forma que los gradientes calculados se emplearán posteriormente para actualizar los pesos de las conexiones sinápticas.

Redes Neuronales Recurrentes (*RNN*)

Las redes neuronales convencionales (véase por ejemplo los *MLP*, las redes neuronales de base radial, etc.) están diseñadas para datos multidimensionales cuyos ejemplos son independientes unos otros (no tienen en cuenta estados anteriores). Sin embargo, existen datos que contienen dependencias de valores previos como los datos de series temporales, biológicos o cadenas de texto [2].

Las redes neuronales recurrentes pueden ser empleadas en este tipo de datos y, además, a diferencia de estas recibe y procesa las entradas en el orden de llegada y las trata de igual manera que los instantes anteriores.

Además, en lugar de un número variable de entradas, contienen un número variable de celdas (operaciones sobre los datos de entrada, para

obtener las predicciones), y cada una de estas se corresponde con un instante de tiempo, lo que permite a cada una interactuar con otras más cercanas a la salida [2].

Cada una de las celdas realiza operaciones con dos entradas y dos salidas que se calculará en cada instante de tiempo t como:

$$\begin{aligned} f: \quad & \mathbb{R}^{m+n} \to \mathbb{R}^{n+l} \\ & (x_t, s_{t-1}) \mapsto (y_t, s_t) \end{aligned} \tag{3.5}$$

Donde x_t es el vector de entrada de dimensión m a la celda en el instante t, y_t es el vector de salida de dimensión n y s_t es el vector de dimensión l que representa el estado actual de la red. Estas celdas compartirán los parámetros internos, por lo que la diferencia entre un instante u otro será las entradas a la celda.

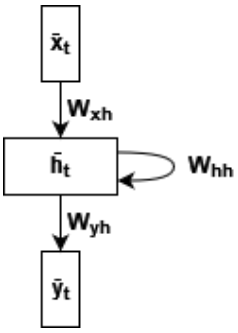

Figura 3.3: Red Neuronal Recurrente.

En la Figura 3.3 puede observarse la representación de una celda de una red recurrente. En esta puede verse cómo dada una secuencia en el instante t, se produce una salida y que generará una secuencia en el tiempo compartiendo parámetros (pesos sinápticos) como puede apreciarse en la Figura 3.4.

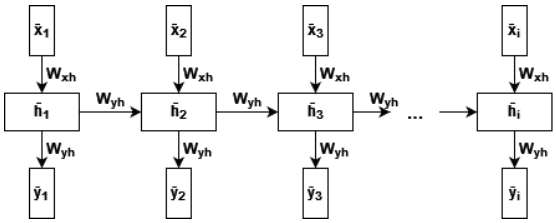

Figura 3.4: Representación de la RNN en el tiempo.

La salida del modelo en un instante de tiempo se calculará, de esta forma cómo [11]:

$$y_t = f(W_{yx}x_t + U_{yy}y_{t-1} + b) \tag{3.6}$$

Donde W_{yx} son los pesos sinápticos en el instante actual en la celda, mientras que U_{yy} se trata de los pesos de la conexión recurrente, es decir, depende de la salida del instante anterior. La primera matriz, de esta forma, tendrá un tamaño de $|x| \cdot |y|$, mientras que la segunda será de tamaño $|y| \cdot |y|$. De manera similar $|b| = |y|$, tratándose este del vector de sesgo.

A diferencia del modelo expuesto con anterioridad, las redes neuronales recurrentes no pueden aplicar el algoritmo de retropropagación (*Backpropagation*) directamente, puesto que en este caso las celdas de instantes posteriores se encontrarán directamente conectadas con las anteriores. En este caso se emplea una adaptación conocida como *Backpropagation through time*, que consiste en "desenrollar" la red recurrente, convirtiéndola en una red *feed-forward* y posteriormente aplicar el algoritmo de retropropagación, teniendo en cuenta que dependiendo de los instantes que se consideren como salidas de la red el gradiente puede afectar a un mismo parámetro de la red de forma repetida [11].

LSTM

Las redes neuronales recurrentes tradicionales pueden encontrar dificultades en capturar y recordar información a largo plazo debido al problema denominado *desvanecimiento del gradiente*, que ocurre cuando los gradientes disminuyen excesivamente a medida que se retropropagan a través del tiempo, produciendo problemas en el aprendizaje a largo plazo.

Para abordar estos problemas, *LSTM* introduce una estructura de memoria adicional denominada celda de memoria que permite al modelo recordar información durante largos periodos de tiempo. Estos modelos emplearán para ello puertas que controlan el flujo de información dentro de estas.

Cada celda en *LSTM* cuenta con tres puertas: la puerta de entrada *Input Gate*, la puerta de salida (*Output Gate*) y la puerta de olvido (*Forget Gate*). La primera decide cuánta información nueva debe debe agregar a la memoria, la segunda cuánta transmitir como salida y la tercera cuánta información antigua se debe de descartar.

En las celdas del modelo se calculan 3 funciones:

$$f_t = \sigma(W_f x_t + U_f y_{t-1} + b_f) \tag{3.7}$$

$$i_t = \sigma(W_i x_t + U_i y_{t-1} + b_i) \tag{3.8}$$

$$o_t = \sigma(W_o x_t + U_o y_{t-1} + b_o) \tag{3.9}$$

La función f_t se corresponde con la puerta de olvido (*Forget Gate*), mientras que i_t con la compuerta de entrada (*Input Gate*) y o_t con la de salida (*Output Gate*)

De esta manera, cada una de estas empleará dos matrices de pesos diferentes (W y U) en cada caso, que se compartirán en distintos instantes de tiempo.

Con los valores de las compuertas, posteriormente se calcula un candidato de la cantidad de información a añadirse:

$$\widetilde{c}_t = tanh(W_c x_t + U_c y_{t-1} + b_c) \tag{3.10}$$

Finalmente el estado de la red se actualiza siguiendo la ponderación:

$$c_t = f_t \odot c_{t-1} + i_t \odot \widetilde{c}_t \tag{3.11}$$

Una vez calculado el nuevo estado, se emplea la puerta de salida, junto con la tangente hiperbólica del nuevo estado oculto de la celda para obtener la nueva salida de la celda:

$$y_t = o_t \odot tanh(c_t) \tag{3.12}$$

En la Figura 3.5 puede observarse el diagrama con las operaciones mencionadas para producir la salida de la celda en el modelo *LSTM*.

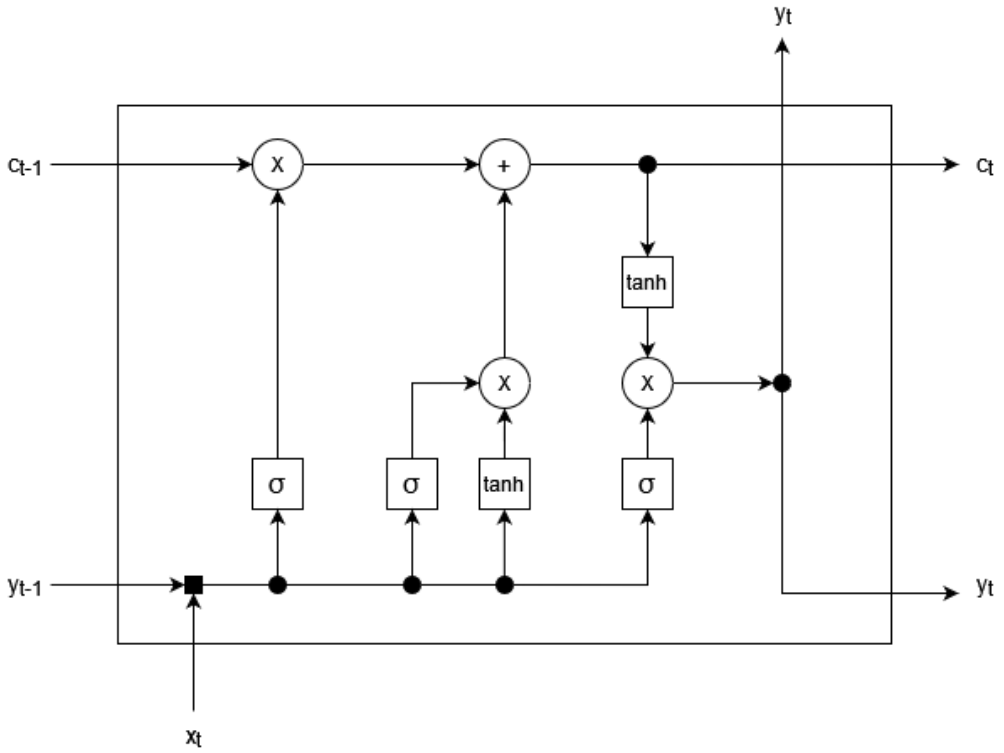

Figura 3.5: Celda de memoria de LSTM

GRU

A diferencia de *LSTM*, este modelo contará con dos compuertas en lugar de tres. Estas serán la puerta de *reset* y la de actualización.

Al igual que antes, los valores en la salida de la puerta se calculan de forma similar, con matrices de pesos diferentes en cada caso. La puerta de actualización se calcula entonces como:

$$z_t = \sigma(W_z x_t + U_z y_{t-1} + b_z) \tag{3.13}$$

De forma análoga la puerta de *reset* se obtiene con la fórmula:

$$r_t = \sigma(W_r x_t + U_r y_{t-1} + b_r) \tag{3.14}$$

De nuevo se calcula la información candidata a añadirse a la memoria:

$$\widetilde{c}_t = tanh(W_c x_t + U_c(r_t \odot y_{t-1}) + b_c) \tag{3.15}$$

Por último, se calcula el nuevo estado de la celda empleando una media ponderada entre el estado anterior y la nueva información:

$$c_t = (1 - z_t) \odot y_{t-1} + z_t \odot \widetilde{c}_t \tag{3.16}$$

Los modelos *GRU* se contemplan como una simplificación de *LSTM*, en la que se fusionan los estados con la respuesta de la red y se fuerza para que estos sean una media ponderada, llevando a obtener resultados similares que con las redes mencionadas [11].

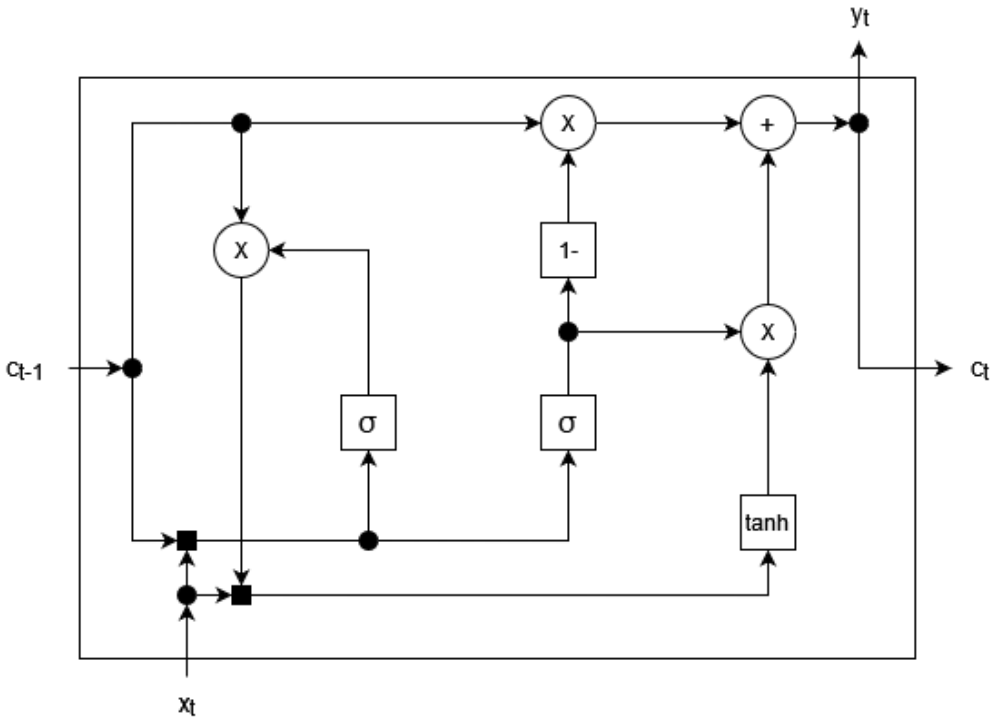

Figura 3.6: Celda de memoria de GRU

Técnicas y herramientas

4.1. Metodologías

Scrum

La metodología *Scrum* es un marco de trabajo de desarrollo ágil de software iterativo (dividido en líneas de tiempo denominados *sprints*) e incremental, que surge como antítesis a la gestión de proyectos predictiva (centrada en la planificación, presupuestos y los plazos de entrega) [12, 13].

4.2. Control de versiones

El control de versiones se realizará mediante Git puesto que a diferencia de otras herramientas de control de versiones, provee de una administración del repositorio distribuida en lugar de centralizada, lo que permite a los desarrolladores tener un clon del repositorio en su equipo local [14].

4.3. Alojamiento del repositorio

Para el *hosting* del repositorio se ha barajado emplear: GitHub, Bitbucket, GitLab.

Se ha optado por emplear GitHub debido a la facilidad de uso que ofrece y las múltiples funcionalidades disponibles, sumado a la destreza adquirida durante el desarrollo del grado con esta plataforma de alojamiento de repositorios.

4.4. Gestión del proyecto

Para la gestión del proyecto se ha considerado emplear ZenHub, por su integración completa con GitHub.

Esta herramienta proporciona un *canvas* que permite gestionar las tareas (*issues*) en el *sprint* ordenándolas en columnas según sea su estado y priorizandolas según se encuentren en esta, además de permitir una estimación de su duración, asignar su desarrollo a uno o más desarrolladores e indicar el *milestone* y el *srpint* al que pertenecen las tareas.

4.5. Comunicación

La comunicación durante desarrollo del proyecto se realiza via e-mail, debido a la rapidez, versatilidad y facilidad de la comunicación, además de la familiaridad de las partes con el medio de comunicación.

Para las tutorías telemáticas se escogió emplear Microsoft Teams por la existencia de forma predefinida de una cuenta de usuario de la Suite Office de Microsoft proporcionada por la Universidad.

4.6. Entorno de desarrollo

Debido a la flexibilidad que aporta Visual Studio Code (editor de texto bajo licencia MIT desarrollado por Microsoft [15]) gracias a el entorno proporcionado de forma nativa y a las extensiones disponibles de Microsoft y terceros, se ha optado por emplearlo para el desarrollo de los scripts de Python, los Python Notebooks, la documentación del repositorio en Markdown y la realización de la memoria en LaTeX.

En cuanto al intérprete Python se ha empleado Python 3.9.13 mediante un entorno virutal que permite instalar las dependencias sin modificar el entorno base.

En lo referente a la compilación de los ficheros LaTeX, se ha optado por emplear MikTex debido a la facilidad de actualización e instalación de dependencias. Para la integración del compilador de LaTeX con Visual Studio Code se emplea LaTeX Workshop, puesto que permite la compilación automática al guardar los cambios en los ficheros sin realizar ninguna configuración explícita.

4.7. Creación de diagramas

Para la creación de diagramas en la memoria y anexos se ha empleado diagrams.net (anteriormente denominada draw.io), puesto que se trata de una herramienta gratuita y de código abierto que puede emplearse de forma *on-line* [16].

Además permite exportar a diferentes formatos como PNG, JPEG, SVG y PDF.

4.8. Librerías

Pandas

Pandas es una herramienta de análisis de datos escrita en Python que ofrece estructuras de datos y operaciones para manipular tablas numéricas y series temporales [17].

Tensorflow

Tensorflow es una librería de código abierto para aprendizaje automático desarrollada por Google [18].

En concreto se ha optado por emplear su *API*: *Keras*, debido a su modularidad y extensibilidad [19].

Matplotlib

Matplotlib es una librería para generar gráficos bidimensionales en Python a partir de diferentes estructuras de datos como listas [20].

Json

El módulo JSON de Python proporciona una *API* (interfaz de programación de aplicaciones) para manejar objetos en formato JSON (*JavaScript Object Notation*) [21].

Os

El módulo OS de Python proporciona una manera portable de emplear la funcionalidad del sistema operativo [22].

Statsmodel

Módulo que proporciona diferentes clases y funciones de distintos modelos estadísticos [23].

4.9. Restful API

Para obtener los datos meteorológicos de la zona en la que se situa el viñedo y poder realizar la comparación con las muestras obtenidas en el pluviómetro desplegado se han estudiado diferentes opciones:

- OpenWeatherMap
- Weatherbit
- AEMET OpenData

Todas las opciones requieren de una petición al servidor mediante HTTPS o HTTP, sin embargo, en el primer caso a pesar de que se muestran los datos en una frecuencia de una hora, el limitante se encuentra en que solo permite recuperar hasta un año atrás. En el segundo caso los requerimientos se cumplen al completo, pero tras contactar con el soporte de la *API* no hay respuestas que permitan el acceso al *endpoint* correspondiente.

Finalmente la última opción proporciona datos históricos sin límite, la contraparte es que la estación meteorológica consultada se encuentra en Aranda de Duero y se tratan de datos diarios.

Aspectos relevantes del desarrollo del proyecto

En este apartado se describirán los aspectos más importantes surgidos a partir del desarrollo del proyecto, estos incluyen las complicaciones encontradas, así como las decisiones tomadas para solucionar todos los problemas surgidos y sus implicaciones.

La sección se orientará siguiendo una estructura parecida a la introducida en *Conceptos teóricos*, es decir, desarrollando los aspectos relevantes según la fase del proyecto en la que se produjo.

5.1. Inicio del Proyecto

Una de las motivaciones que se tuvo en cuenta para el desarrollo del proyecto fue la cercanía del tema tratado con el lugar que se desarrolló. Y es que la provincia de Burgos se encuentra en la denominación de origen "Ribera del Duero", una de las regiones que cuenta con algunos de los mejores vinos del mundo [24].

Otro impulso que llevó a realizar el Trabajo Final de Grado con este tema fue, de igual forma, la importancia del vino en la familia, pues el sustento de parte de esta es gracias a la viticultura, además de la historia e importancia de la cultura del vino en el todo el territorio español.

Tras contactar con los docentes, se proporcionó el conjunto de datos empleado en el proyecto. En estas conversaciones iniciales se consideró que estos contaban con ciertas características que debían tenerse en cuenta: en primer lugar, los presentaban valores faltantes debidos a la posible

desconexión de los sensores de la red, lo que provocaba que estos dejaran de enviar datos al servidor. Por otro lado, las labores del campo podían haber afectado a la calidad de estos por la extracción de las sondas de los sensores de sus posiciones originales.

Además, el pluviómetro instalado en un orígen se encontraba anclado de forma débil, lo que propiciaba que el viento produjera lecturas erróneas en el movimiento del balancín empleado en el registro de las precipitaciones.

Tras estas reuniones se inició el proceso de la comprensión del negocio, es decir, se comenzó con la definición de los objetivos del proyecto y los requisitos. Simplificando lo mencionado en la introducción y los objetivos, la finalidad era analizar los datos proporcionados (centrándose en la temperatura y humedad) realizando las visualizaciones pertinentes y encontrar modelos predictivos que intenten anticipar la evolución a corto plazo empleando diferentes técnicas como las Redes Neuronales Artificiales.

5.2. Fase de preparación de los datos

En la fase de preparación se determinó que los datos proporcionados contaban con diferentes situaciones. En algunos de los sensores la calidad de estos era bastante adecuada a pesar de contar con valores faltantes, en otros, en cambio, existían numerosos problemas.

Por un lado, contaban con valores extremos que debían tenerse en cuenta para que los modelos que se desarrollasen no encontraran problemas al generalizar, desviándose de ese propósito por el posible ruido introducido. Para solucionar este problema se empleó el mecanismo de detección de *outliers* mediante el rango intercuartílico, sustituyendo los datos identificados como extremos con la mediana del día en el que se realizó la muestra.

Por otro lado, algunos de los sensores contaban con valores matemáticamente correctos (dentro de límites lógicos de las diferentes variables) pero que contextualmente eran incorrectos. Un ejemplo de esto eran las lecturas artificiales de distintas variables como la temperatura cuando el nivel de la batería de los sensores caía por debajo de 3,3V, lo que probocaba un desbordamiento de la variable en el controlador correspondiente.

Ante esta situación se decidió eliminar aquellas muestras cuyo nivel de batería se encontrara por debajo del mencionado y por encima de los 4,1V (el máximo). Sin embargo, tras realizar esta operación se observó que ciertas muestras continuaban siendo contextualmente incorrectas arrojando datos inverosímiles (bajadas de temperatura y/o humedad extremas en periodos

de tiempo cortos) debidas en gran medida, como se confirmó posteriormente, a la extracción de las sondas de la situación original, por lo que se acordó con los docentes la eliminación de estos valores que continuaban siendo erróneos.

El proceso se realizó a mano teniendo en cuenta las situaciones localizadas con anterioridad y con las nuevas necesidades, marcando en los diferentes ficheros la validez de la entrada (cabe destacar que la recuperación de estos datos no era viable por las condiciones en la que se encontraban, puesto que el lapso temporal era demasiado elevado).

Además de estas condiciones, algunos de los sensores contaban con muestras con discontinuidades inexplicables mediante los datos, por lo que se acordó el intento de la recuperación de estos. Para ello, se observaron los muestreos de los sensores geográficamente más cercanos al afectado, de forma que se pudiera conocer de manera más precisa cúal de las series era la que contaba con la lecura correcta y acutar en consecuencia.

A pesar de que se haya podido introducir algún sesgo debido a la modificación de los valores plasmados en los muestreos, los datos reales serán más cercanos a los nuevos valores asignados que a las lecturas anteriores, es decir, aún si estos fueran incorrectos serían menos incorrectos que anteriormente.

Ante la premura de los plazos del proyecto, realizar de nuevo el muestreo de los datos suficientes para crear un modelo predictivo eficiente no era viable, por lo que se tuvo que trabajar con la cantidad proporcionada.

5.3. Fase de creación de modelos

Tras realizar una investigación preliminar acerca de los modelos con mayor precisión en la regresión de series temporales, se llegó a la conclusión de emplear redes neuronales recurrentes con memoria como primer objetivo de modelado, puesto que sobre datos meteorologicos contaban con un buen rendimiento.

Por ese motivo, se escogió utilizar las redes neuronales con memoria a corto y medio plazo como son *GRU* y *LSTM*, empleando los mismos hiperparámetros y, de esta manera, poder realizar una comparativa para tener una idea de cómo se ajustan estos modelos a las diferentes funciones objetivo.

Además, otra opción a emplear fue la tradicional *MLP* para poder observar cómo la linealidad de este modelo (no tiene en cuenta instantes anteriores) impide obtener unos resultados aceptables en comparación de sus contrapartes recurrentes.

El primer reto consistía en preparar los datos de entrada a la red neuronal y los correspondientes salidas de forma que estos fueran muestras homogéneas sin saltos temporales que impidiearan a los modelos ajustarse adecuadamente.

En primer lugar, en un espacio de 10 minutos se podían encontrar 2 o 3 muestras según fuera el *offset* del controlador de muestreos de los sensores. Además, como en la fase de preparación de los datos se realizaron descartes, se generaron, de esta manera, saltos temporales que podían afectar a la predicción de la tendencia.

Una solución fue recorrer el conjunto de datos del sensor con una ventana deslizante del tamaño equivalente a las instancias de entrada (establecido por parámetro) y el dato de salida (se realizó una aproximación preliminar realizando una única predicción, que posteriormente se modificó para que los modelos fuesen capaces de realizar varias), de esta forma en ventanas en las que no habían datos suficientes para la entrada se descartaban.

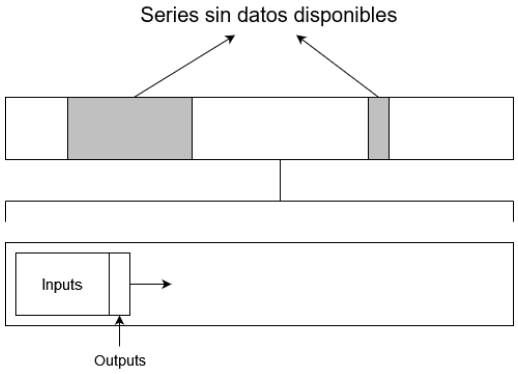

Figura 5.1: Esquema del proceso de preparación de datos

Cabe destacar que inicialmente se emplearon todos los datos para entrenamiento y validación, no estableciendo el conjunto de test, lo que fue modificado más tarde añadiendo el tercer conjunto, permitiendo reducir el sesgo en la evaluación del modelo, pues, a diferencia de las instancias de validación, se tratan de datos que no han sido empleados por el modelo tanto para ser entrenado como validado.

Tras la preparación de los datos de entrada de la red se comenzó a realizar el modelo *GRU*, de forma que se debían establecer sus capas y las celdas en estas. Posteriormente se realizaron los modelos *LSTM* y *MLP*. Para agilizar la construcción y verificar su funcionamiento se estableció inicialmente 3 capas en todos ellos, con 36, 16 y 8 unidades sucesivamente.

A diferencia de los dos primeros, *MLP* requería de una ligera modificación, puesto que no aceptaba secuencias, por lo que se estableció una neurona de entrada por cada valor en la serie de tiempo y de igual forma una neurona en la capa de salida por cada predicción.

5.4. Fase de evaluación de modelos

En la fase de evaluación se estableció realizar pruebas sobre los modelos modificando ciertos hiperparámetros como el número de capas ocultas, el número de unidades (celdas o neuronas dependiendo del contexto) por capa y ratio de aprendizaje.

De esta forma, los modelos empleados tendrán las especificaciones estándar en el caso de *GRU* y *LSTM*, modificando la función de activación de las neuronas en las capas ocultas en *MLP* por una función *ReLU* (Figura 5.2), que aportará la no linealidad al modelo, lo que presumiblemente mejoraría el rendimiento concreto con los datos empleados.

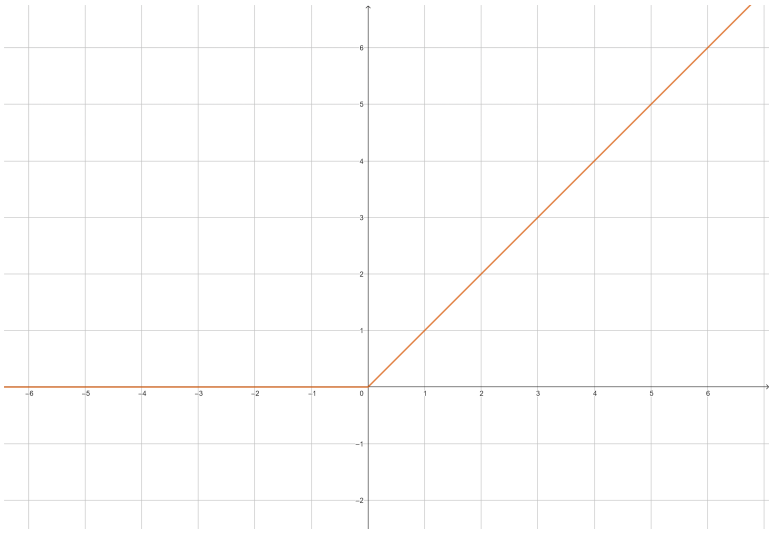

Figura 5.2: Función ReLU

Se realizarán, entonces, combinaciones con diferentes valores:

- Capas: 1, 2 y 3 capas.
- Tamaño de capa: 8, 16 y 32 celdas/neuronas.
- Ratio de aprendizaje: 0.1, 0.01 y 0.001.

Trabajos relacionados

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

Conclusiones y Líneas de trabajo futuras

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

Bibliografía

[1] I. H. I. H. Witten, *Data mining : practical machine learning tools and techniques*, fourth edition. ed. Amsterdam: Elsevier, 2017.

[2] C. C. Aggarwal, *Neural networks and deep learning : a textbook*, 1st ed. Cham: Springer International Publishing, 2018.

[3] J. Piqueras, *Historia de la vid y el vino en España edades Antigua y Media*. Valencia: Univeritat de València, 2014, ch. 1.

[4] Ministerio de Agricultura, Pesca y Alimentación, "El cultivo de la vid en españa," [Online; Accedido 20-marzo-2023]. [Online]. Available: https://www.mapa.gob.es/es/ministerio/archivos-biblioteca-mediateca/archivos/Cultivo.aspx

[5] Mundi-Prensa, *Los parásitos de la vid estrategias de protección razonada*, 5th ed. Madrid: Mundi-Prensa, 2004, ch. 1, pp. 17–18.

[6] ——, *Los parásitos de la vid estrategias de protección razonada*, 5th ed. Madrid: Mundi-Prensa, 2004, ch. 3, p. 66.

[7] I. Z. L. Zúñiga, *Meteorología y climatología*, Ed. digital act.: sept. de 2021 ed., ser. Grado. Madrid: Universidad Nacional de Educación a Distancia, 2021, ch. 11, pp. 194, 216.

[8] G. Hernández, "Repositorio del Trabajo Fin de Grado 'Análisis de datos de temperatura y huemedad de suelo procedentes de sensores IoT desplegados en un viñedo' por Gabriel Hernández Vallejo." [Online]. Available: https://github.com/GabiHV/TFG22-23

[9] J. Hernández Orallo, *Introducción a la minería de datos*. Madrid: Pearson Educación, 2004, ch. 4.

[10] Wikipedia, "Filtro de hodrick-prescott — wikipedia, la enciclopedia libre," 2023, [Internet; descargado 3-febrero-2023]. [Online]. Available: https://es.wikipedia.org/w/index.php?title=Filtro_de_Hodrick-Prescott&oldid=149041721

[11] A. Bosch Rueé, *Deep learning : principios y fundamentos*, primera edición en lengua castellana ed., ser. Manuales (Tecnología). Barcelona: Editorial UOC, 2019 - 12??

[12] Wikipedia contributors, "Scrum (software development) — Wikipedia, the free encyclopedia," 2023, [Online; accessed 24-March-2023]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Scrum_(software_development)&oldid=1144400875

[13] M. Palacio, *Scrum Master*. Iubaris Info 4 Media SL, 2021, ch. 1, p. 11.

[14] S. F. Conservancy, "About - git," 2023. [Online]. Available: https://git-scm.com/about/distributed

[15] Wikipedia contributors, "Visual studio code — Wikipedia, the free encyclopedia," 2023, [Online; accessed 24-March-2023]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Visual_Studio_Code&oldid=1145496466

[16] ——, "Diagrams.net — Wikipedia, the free encyclopedia," 2023, [Online; accessed 30-May-2023]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Diagrams.net&oldid=1142089111

[17] Wikipedia, "Pandas (software) — wikipedia, la enciclopedia libre," 2023, [Internet; descargado 24-marzo-2023]. [Online]. Available: https://es.wikipedia.org/w/index.php?title=Pandas_(software)&oldid=148434974

[18] ——, "Tensorflow — wikipedia, la enciclopedia libre," 2021, [Internet; descargado 24-marzo-2023]. [Online]. Available: https://es.wikipedia.org/w/index.php?title=TensorFlow&oldid=139184038

[19] ——, "Keras — wikipedia, la enciclopedia libre," 2022, [Internet; descargado 25-marzo-2023]. [Online]. Available: https://es.wikipedia.org/w/index.php?title=Keras&oldid=148156962

[20] ——, "Matplotlib — wikipedia, la enciclopedia libre," 2022, [Internet; descargado 24-marzo-2023]. [Online]. Available: https://es.wikipedia.org/w/index.php?title=Matplotlib&oldid=146868590

[21] P. S. Foundation, "json — json encoder and decoder," 2023. [Online]. Available: https://docs.python.org/3/library/json.html#module-json

[22] ——, "os — miscellaneous operating system interfaces," 2023. [Online]. Available: https://docs.python.org/3/library/os.html?highlight=os#module-os

[23] J. T. Josef Perktold, Skipper Seabold, "statsmodels 0.14.0," 2023. [Online]. Available: https://www.statsmodels.org/stable/index.html

[24] ABC.es, "Dos vinos de la ribera del duero entre los '7 magníficos' del mundo," Sep 2021. [Online]. Available: https://www.abc.es/viajar/noticias/abci-vinos-ribera-duero-entre-7-magnificos-mundo-202109160047_noticia.html