Predicción de Series Temporales con Redes Neuronales

Autor: Ing. Ariel Alejandro Fierro
Director: Dr. Franco Ronchetti

Trabajo Final presentado para obtener el grado de Especialista en Inteligencia de Datos Orientada a Big Data.

Facultad de Informática
Universidad Nacional de La Plata
Argentina
Junio de 2020

Índice

1. Introducción

Una serie temporal es una sucesión ordenada de datos u observaciones medidos en el tiempo. Una de las utilidades más importantes de las series temporales es su análisis para la predicción de la variable medida.

El aprendizaje automático (Machine Learning) es una rama de la inteligencia artificial que tiene como objetivo proporcionar a las computadoras la capacidad de aprender sin ser programadas. En lugar de ser los programadores los creadores de las reglas para el procesamiento de datos, las computadoras aprenden modelos que representan los datos. Estos modelos son posteriormente utilizados para originar respuestas a partir de nuevos datos.

Las redes neuronales son un campo dentro del aprendizaje automático en constante desarrollo, que en los últimos tiempos ha ganado una enorme popularidad gracias a los resultados prometedores que se obtienen en campos de aplicación de lo más diversos. Uno de particular interés es la predicción de series temporales. En una serie temporal cada dato tiene una dependencia temporal con las muestras anteriores y posteriores. Esto hace que se requiera de un análisis particular del caso para poder lograr una predicción a partir del pasado. Ejemplos de modelos predictivos basados en redes neuronales pueden encontrarse en finanzas [2,1], comercio electrónico [6], mercados de capitales [3], variables macroeconómicas [4], salud [7,8], procesamiento de señales, meteorología [9, 22, 23], reconocimiento de voz [11] y control de tráfico [10].

1.1. Motivación

Debido al interés creciente por esta tecnología hay una gran cantidad de reportes en la literatura en los que se realizan experimentos sobre distintos conjuntos de datos verificando la performance de las redes neuronales para tareas predictivas. Los estudios coinciden en que los resultados son muy prometedores pero tal como se indica en [12,13] las conclusiones obtenidas al comparar éste tipo de métodos con los estadísticos son inconsistentes entre sí.

De acuerdo a [13], un problema con la literatura asociada a las predicciones de series temporales con aprendizaje automático es que la mayoría de los estudios publicados proveen predicciones aclamando precisiones satisfactorias sin compararlas con métodos estadísticos o inclusive simples puntos de referencia. Este hecho crea altas expectativas con respecto a la capacidad predictiva de las técnicas de aprendizaje automático, pero sin una prueba empírica de que este sea el caso, habilitando conclusiones que podrían no ser sostenibles. Por ejemplo en [13] se concluye que los métodos basados en Redes Neuronales no son tan buenos como los estadísticos, mientras que en [17,18,19,20] se resuelve lo contrario. Por otro lado, las soluciones que obtuvieron los mejores resultados en las competencias abiertas de comunidades online referentes en ciencia de datos como Makridakis [14,15] surgieron a partir de la aplicación en forma combinada de ambos tipos de métodos como en [16].

La motivación por la realización del presente Trabajo Final Integrador surge a partir de las posiciones encontradas en la literatura las cuales ameritan su investigación y análisis, siendo una de las contribuciones del mismo la obtención de una opinión objetiva y fundamentada al respecto.

1.2. Objetivos

El objetivo general del presente Trabajo Final Integrador es comparar distintos métodos de aprendizaje automático basados en redes neuronales con el método estadístico vectorial autorregresivo (VAR, por su sigla en inglés) para la predicción de series temporales multivariadas.

Los objetivos específicos a realizar para alcanzar el objetivo general son los siguientes:

- Analizar el desempeño del método VAR para la predicción de una serie temporal multivariada.
- Analizar el desempeño de distintos métodos de aprendizaje automático para la predicción de una serie temporal multivariada.
- Identificar posibles causas de las diferencias en la literatura respecto de la precisión de los distintos modelos predictivos.

2. Conceptos preliminares

Series temporales - Una serie temporal puede ser definida como un conjunto temporalmente ordenado de observaciones que puede ser discreta o continua. Comúnmente tienen una frecuencia fija, lo que significa que las observaciones ocurren en intervalos fijos de tiempo.

El análisis de series temporales es el estudio de series temporales de datos con el objetivo de extraer parámetros relevantes o características de ella. Estos parámetros y características pueden ser luego utilizados para generar un modelo matemático que describa la serie y pueda ser utilizado para realizar predicciones.

De acuerdo a la cantidad de variables o características que la serie temporal contenga se considera univariada para el caso de una variable, o multivariada para el caso de múltiples variables. Dependen de esta fundamental característica los pasos a seguir para su análisis y predicción.

Es importante destacar que una variable de una serie de datos temporales puede no representar solamente valores medidos u observados en el tiempo. Por ejemplo a partir del procesamiento de una o más variables originales de la serie podría obtenerse una nueva variable aportando información nueva o reforzando información preexistente en la serie [33, 34]. A esto se lo conoce como ingeniería de características.

Estacionariedad - Una serie de datos temporal es estacionaria cuando sus propiedades estadísticas no dependen del tiempo al momento en que la serie es observada. Una forma de ver la condición de estacionariedad es mediante la correlación de sus datos. Si no están correlacionados entre sí entonces la serie es estacionaria. Series temporales con tendencias crecientes o decrecientes, con patrones estacionales o cíclicos, o con varianza variable no son estacionarias. La tendencias y los patrones estacionales afectarán los valores de la serie temporal en momentos diferentes. Por ejemplo, una serie de ruido blanco (señal aleatoria cuyos valores son temporalmente independientes) es estacionaria. La estacionariedad o no estacionariedad de los datos puede ser difícil de verificar a partir de un gráfico de los mismos únicamente. Una serie temporal con comportamiento cíclico pero sin tendencia puede ser estacionaria. Esto es así porque los ciclos no tienen una longitud fija, no es posible determinar cuándo serán los picos y valles del ciclo. En general una serie temporal estacionaria no tiene patrones predecibles en el largo plazo.

Es posible convertir una serie temporal en estacionaria por medio de la aplicación de las transformaciones correctas, lo cuál es muy importante para el análisis de series temporales, ya que la mayoría de los métodos predictivos estadísticos están diseñados para trabajar con series temporales estacionarias [31]. Esto permite realizar una predicción más precisa y confiable.

Correlación lineal - La correlación lineal entre dos variables es la relación estadística entre ellas. La correlación puede cuantificarse en un valor que, en el caso de ser positivo, significa que ambas variables cambian en la misma dirección.

También puede ser negativo, lo que significa que ambas variables cambian en direcciones opuestas, o, puede ser cero, lo que significa que ambas variables no están relacionadas linealmente [32].

Autocorrelación - La autocorrelación de las observaciones de una serie temporal es la correlación de las observaciones de la serie con las observaciones de una copia de la misma serie desplazada temporalmente. A la función que permite calcular dicha propiedad entre observaciones se le llama función de autocorrelación (ACF, por su sigla en inglés). Representa el gráfico de la cuantificación, en un rango de valores entre 1 y -1, de la autocorrelación de las observaciones. El gráfico de esta función incluye intervalos de confianza establecidos por defecto en un 95 % con límites en $\pm 1{,}96\sqrt{n}$ [32], cuyo significado es que los valores de correlación que estén por fuera de dicho intervalo corresponden a una correlación y no a una anomalía estadística. Esta regla depende de una muestra de tamaño lo suficientemente grande y una varianza finita para el proceso. En la figura 1 se muestra a modo de ejemplo la función de autocorrelación de las primeras 100 observaciones de la variable de temperatura. El intervalo de confianza se observa sobre el eje horizontal en color azul. A partir del gráfico se puede deducir que dichas observaciones están correlacionadas.

En el presente Trabajo Final Integrador se utiliza la función $plot\_acf()$ de la librería Statsmodels (ver Anexo), la cuál recibe como parámetro una serie temporal de una sola variable y la cantidad de pasos temporales para graficar la función de autocorrelación asociada.

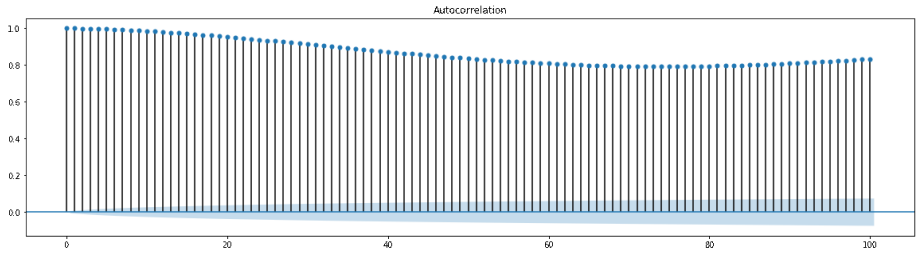

Figura 1: Función de autocorrelación de las primeras 100 observaciones de la variable de temperatura.

Para el análisis de series temporales, la autocorrelación proporciona información sobre cómo están relacionados los datos, las periodicidades y sus frecuencias características. La función ACF de una función periódica tiene la misma periodicidad que el proceso original. A su vez, si las observaciones de una serie temporal no muestran ningún tipo de correlación, significa que dicha serie no es predecible.

Diferenciación - La diferenciación es una de las transformaciones más comunes y convenientes para convertir una serie temporal no estacionaria en estacionaria. Esto se logra computando las diferencias entre observaciones consecutivas. Transformaciones como las logarítmicas pueden ayudar a estabilizar la varianza de una serie temporal. La diferenciación en cambio puede ayudar a estabilizar

la media de una serie temporal, por consiguiente reduciendo la tendencia y los patrones periódicos.

La diferenciación de la serie en términos prácticos es la sustracción del valor actual al valor siguiente. Si Y es el valor de una observación al momento 't', entonces la primera diferenciación de la serie puede escribirse como $Y'_t = Y_t - Y_{t-1}$. La serie diferenciada tendrá solamente $t-1$ valores, ya que no es posible calcular Y'_1 para la primera observación. Si la primera diferenciación no convierte a la serie en estacionaria, es posible realizar una segunda diferenciación. Este procedimiento se puede repetir en caso de ser necesario. En la práctica, sin embargo, no es usual ir más allá de una segunda transformación. La diferenciación afecta a la correlación entre observaciones, por eso si se diferencia de más, se elimina completamente dicha correlación y la serie se vuelve imposible de predecir.

Una manera de determinar de forma cuantitativa si una serie temporal es estacionaria es utilizando tests estadísticos llamados 'Tests de Raíz Unitaria' [25,35]. Son tests de hipótesis estadísticas de estacionariedad utilizados en el análisis de series temporales. Permiten determinar si es necesario aplicarle transformaciones a la serie. Hay múltiples implementaciones de Tests de Raíz Unitaria como:

- Augmented Dickey Fuller test (ADF).
- Kwiatkowski-Phillips-Schmidt-Shin (KPSS).
- Philips Perron test (PP).

Para el análisis del presente Trabajo Final Integrador se utiliza el test de Kwiatkowski-Phillips-Schmidt-Shin (KPSS) [25]. En dicho test, la hipótesis nula afirma que los datos son estacionarios, luego se busca evidencia para rechazar la hipótesis nula. Por convenio suele establecerse que si el valor de probabilidad devuelto por el test es menor al 5 % (0.05) entonces es lo suficientemente improbable que lo observado se deba al azar como para rechazar con una seguridad razonable la hipótesis nula. Por eso, para valores de probabilidad menores que 0.05 se asume que la serie no es estacionaria.

Test de causalidad de Granger - La base detrás del modelo estadístico Vector Autorregresivo indica que las variables que componen la serie temporal multivariada del sistema ejercen algún tipo de influencia entre sí, lo cual hace posible predecir los valores de una variable de la serie utilizando los valores u observaciones pasadas de la misma variable así como de otras.

El test de causalidad de Granger [28] es una forma de investigar la causalidad entre dos variables en una serie temporal de datos, de poner a prueba esta relación antes de generar el modelo predictivo. Es un procedimiento probabilístico cuya hipótesis nula afirma que los valores pasados de una serie temporal X no causan los valores de la otra serie Y. Entonces si el valor de probabilidad devuelto por el test es menor que el valor de significancia 0.05, entonces es posible rechazar la hipótesis nula.

Transformación de Box Cox - La transformación de Box Cox [41] consiste en una familia de funciones de potencia que combinan el objetivo de homogeneizar las varianzas para diferentes valores de la variable predictora, y mejorar la correlación entre las variables. La transformación de Box Cox está dada por las funciones:

$$y = \begin{cases} (x^\lambda - 1)/\lambda, & \text{para } \lambda > 0 \\ \log x, & \text{para } \lambda = 0 \end{cases}$$

donde λ es el parámetro de transformación.

2.1. Métricas para regresión

A la hora de seleccionar una métrica de error es de suma importancia tener en cuenta el campo de estudio, el conjunto de datos y el rol que cumplen los errores. Las distintas métricas de error describen características diferentes de los modelos predictivos y de los datos. Dos de las métricas más utilizadas para problemas de regresión son el error absoluto medio (MAE, por su sigla en inglés) y el error cuadrático medio (MSE, por su sigla en inglés).

MSE - El error cuadrático medio es una métrica simple y común para evaluar un problema de regresión. Se define por medio de la ecuación

$$MSE = \frac{1}{N}\sum(y_i - y_i')^2$$

donde y_i es la salida esperada e y_i' es la predicción del modelo. Para cada punto, calcula el cuadrado de la diferencia entre las predicciones y el valor real y luego promedia esos valores. Cuanto mayor es el valor de la métrica, peor es el modelo. Nunca es negativo, pero su valor es cero para un modelo perfecto. Es útil cuando los datos poseen valores atípicos, ya que dicha métrica los resalta. Su desventaja es que en el caso de obtener malas predicciónes del modelo la elevación cuadrática empeora el problema, sobreestimando lo malo del mismo. Por otro lado, si todos los errores son menores que 1 se obtiene el efecto contrario, se subestima lo malo del modelo.

MAE - El error absoluto medio es calculado como el promedio de las diferencias absolutas entre los valores objetivos y las predicciones. Es utilizado en el presente Trabajo Final Integrador para la evaluación y comparación de los modelos predictivos y como métrica de la función de pérdida de los modelos de redes neuronales. MAE es una puntuación lineal que significa que todas las diferencias individuales son ponderadas igualitariamente en el promedio. Por ejemplo la diferencia entre 10 y 0 será el doble que la diferencia entre 5 y 0. Sin embargo lo mismo no es cierto para MSE. Matemáticamente se calcula utilizando la siguiente ecuación:

$$MAE = \frac{1}{N}\sum|y_i - y_i'|$$

MAE es más robusto, menos sensible a valores atípicos que MSE pero esto no significa que siempre sea mejor usar MAE. Por ejemplo, es más indicado utilizar MSE como métrica si es necesario que los valores atípicos tengan mayor

preponderancia sobre el modelo predictivo. En caso contrario, de ser necesario disminuir su importancia, es más indicado utilizar MAE ya que los valores atípicos de los residuales (diferencia entre el valor actual y la estimación del modelo) no contribuyen tanto al error total como al utilizar MSE. Ambas son métricas de error viables, pero describen en forma diferente los errores predictivos de los modelos.

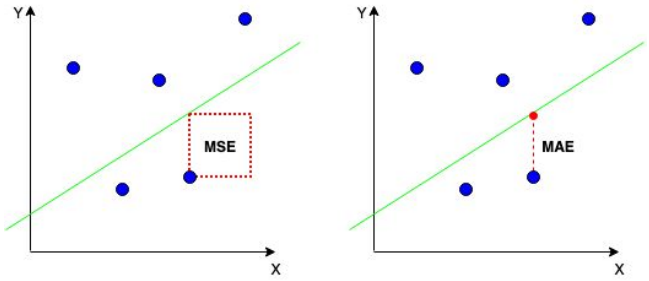

Figura 2: MSE(*izq.*) - MAE(*der.*)

2.2. Criterios de información de Akaike, Bayesiano y de Hannan-Quinn

Los criterios de información de Akaike (AIC), Bayesiano (BIC) y de Hannan-Quinn (HQC) son medidas de la calidad relativa de un modelo estadístico, para un conjunto determinado de datos. Proporcionan un medio para la comparación y selección de un modelo entre varios. Dado un conjunto de datos, varios modelos candidatos pueden ser clasificados de acuerdo a su AIC, BIC y HQC siendo el mejor el que presente valores mínimos para cada criterio. Se definen de la siguiente manera:

$$\begin{aligned} AIC &= 2k - 2\ln L, \\ BIC &= k\ln n - 2\ln L, \\ HQC &= 2k\ln\ln n - 2\ln L, \end{aligned}$$

donde en cada caso k es el número de parámetros en el modelo estimado, n es el número de observaciones, y L es el máximo valor de la función de verosimilitud para el modelo estimado. La práctica habitual es ajustar el modelo utilizando el orden de retraso mejor calificado según uno de los criterios anteriores. La librería *Statsmodels* (ver Anexo) provee un método que devuelve los valores óptimos de retraso para cada criterio a partir de un modelo y un conjunto de datos.

3. Métodos predictivos

La idea de estos métodos es creación de un modelo basado en ecuaciones matemáticas que representen la interacción de las variables involucradas.

3.1. Métodos predictivos estadísticos

Modelo Autorregresivo - El modelo predictivo Autoregresivo (AR) se basa en la idea de que las observaciones pasadas pueden ser utilizadas para predecir observaciones a futuro mediante un proceso, por medio del cual, un valor en un determinado momento puede calcularse en función de los valores de la misma serie en momentos pasados [35]. Los modelos autorregresivos son muy flexibles en cuanto al manejo de un amplio abanico de patrones de series temporales diferentes, pero requieren que la serie temporal sea estacionaria. El modelo más simple Autorregresivo, el AR(1), se describe por medio de la siguiente ecuación:

$$X_t = \alpha_1 + \beta_1 X_{t-1} + \varepsilon_t$$

El valor de la serie al momento t es una función con un valor constante α_1, su valor al momento $t-1$ multiplicado por otra constante $\beta_1 X_{t-1}$ y un error ε_t cuyo valor varía también con el tiempo. Se asume que dicho error posee varianza constante y media 0 (cero). Para la creación de un modelo AR es necesario determinar cuántas observaciones pasadas (denotadas por p) deben incluirse en el sistema. Luego, en términos generales, un modelo AR que depende de los p valores más recientes $AR(p)$ se describe de la siguiente manera:

$$X_t = \alpha_1 + \beta_1 X_{t-1} + \alpha_2 + \beta_2 X_{t-2} + .. + \alpha_p + \beta_p X_{t-p} + \varepsilon_t$$

Para la selección del número de observaciones pasadas a incluir son utilizados los criterios informativos AIC, BIC y HQC.

Modelo Vector Autorregresivo - El modelo VAR es un modelo predictivo estadístico que consiste en la generación de un modelo $AR(p)$ para el caso de una serie temporal de múltiples variables, donde cada variable ejerce una influencia sobre el resto [35]. Una de sus limitaciones es que el ajuste de sus parámetros se hace en forma simétrica con respecto a todas las variables. Al igual que en el caso del modelo AR, el modelo VAR requiere que la serie temporal sea estacionaria. Al tratarse en este caso de series de múltiples variables, significa que cada una de las series correspondientes a los valores de cada variable sea estacionaria. Es considerado un modelo autorregresivo porque cada variable es modelada en función de valores pasados. Como cada variable es utilizada para predecir sus valores a futuro, al igual que del resto de las variables, tendrá un sistema de ecuaciones con una ecuación por variable. Por ejemplo, el sistema de ecuaciones para un modelo VAR(1) de dos variables X_1 y X_2 sería el siguiente:

$$X_{1,t} = \alpha_1 + \beta_{11,1} X_{1,t-1} + \beta_{12,1} X_{2,t-1} + \varepsilon_{1,t}$$
$$X_{2,t} = \alpha_2 + \beta_{21,1} X_{1,t-1} + \beta_{22,1} X_{2,t-1} + \varepsilon_{2,t}$$

Para crear un modelo VAR es necesario determinar cuántas variables tendrá (denotado por K) y cuántas observaciones pasadas (denotadas por p) deben incluirse en el sistema. El número de coeficientes a estimar en un VAR es igual a $K+pK^2$ o $1+pK$ por ecuación. Por ejemplo, para un VAR con $K = 5$ variables y $p = 3$ observaciones pasadas, hay 16 coeficientes por ecuación, lo que implica un total de 80 coeficientes para ser estimados. Cuanto mayor es la cantidad de coeficientes a estimar, mayor es el error de estimación en los resultados del pronóstico. Al igual que en el caso del modelo Autorregresivo, para determinar el número de observaciones pasadas del modelo VAR son utilizados los criterios informativos AIC, BIC y HQC.

Una forma de definir si una variable es útil para predecir la otra es mediante el test de causalidad de Granger.

3.2. Métodos predictivos de aprendizaje automático

El mecanismo de las redes neuronales se inspira en la estructura de una red neuronal biológica. Estas utilizan una serie de unidades de procesamiento interconectadas , llamadas neuronas artificiales, para calcular las entradas recibidas y producir los resultados de una suma ponderada de entradas. El resultado luego es alimentado a una función no lineal, llamada función de activación, para generar el resultado final.

3.2.1. Perceptrón

La neurona artificial es un modelo matemático muy simple compuesto por una o más entradas y una salida. La neurona artificial simplemente activa su salida cuando el total del estímulo recibido supera un determinado umbral. Se asocia muchas veces a una neurona del cerebro humano, pero ese no es su objetivo, sino el de crear algoritmos que puedan modelar distintos problemas. El poder de las redes neuronales proviene de su capacidad para aprender la representación de los datos de entrenamiento y de cómo los relaciona a la salida de la variable a predecir. Por este motivo las redes neuronales aprenden a hacer un mapeo matemáticamente. Esta estructura puede aprender a representar variables en diferentes escalas y combinarlas en variables más complejas.

Un Perceptrón es un modelo de una sola neurona artificial, precursor de una red neuronal que consiste en un conjunto de entradas numéricas, pesos sinápticos asociados a las entradas, una entrada extra denominada *bias*, una regla de propagación que comúnmente es una suma ponderada, una función de activación, y por último una salida numérica. Los pesos sinápticos son coeficientes que parametrizan las capas, y definen la intensidad de la interacción de las entradas sobre las salidas. La función de activación mapea la suma de las señales ponderadas a la señal de salida. Sin una función de activación la salida del Perceptrón consistiría únicamente de dos operaciones lineales, luego éste sería capaz de aprender únicamente transformaciones lineales o afines. Para evitar esta limitación se utiliza la función de activación, lo que le permite al Perceptrón realizar transformaciones no lineales.

Hay varios tipos de funciones de activación y su utilización depende del objetivo de la red y de la capa en la misma. Relu (Rectified Linear Unit) es la función de activación más común en capas intermedias. Relu es una función que convierte en cero los valores negativos, definida por la ecuación $f(z) = \text{máx}(0, z)$. Para los casos donde la red es utilizada para clasificar, la capa final utiliza una función sigmoide, de modo que el resultado de la red pueda ser interpretado como una probabilidad. Una función sigmoide convierte todos los valores de entrada al intervalo 0-1 y se define por medio de la ecuación $f(z) = 1/(1 + e^{-z})$. En los problemas de regresión, como en el caso del presente Trabajo Final Integrador, la última capa no posee una función de activación, sino una función de identidad.

Finalmente el Perceptrón, para obtener su salida o resultado $f(z)$, aplica una función de activación al resultado de la suma ponderada de las entradas más el *bias*.

$$z = \omega_1 x_1 + \omega_2 x_2 + .. + \omega_n x_n + b$$

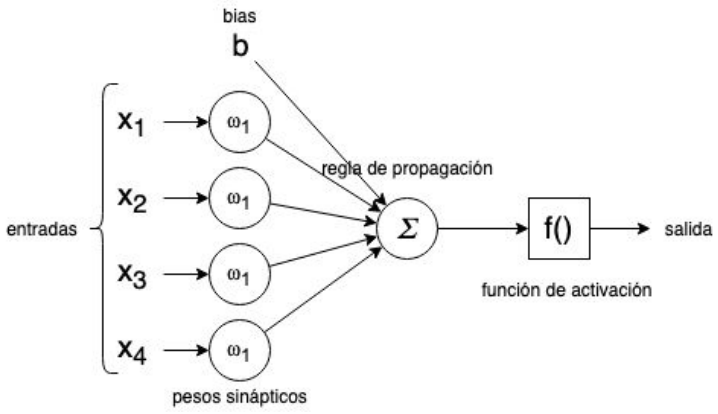

Figura 3: Perceptrón.

Comúnmente las neuronas se organizan en capas, y a su vez, una red neuronal puede estar compuesta por múltiples capas.

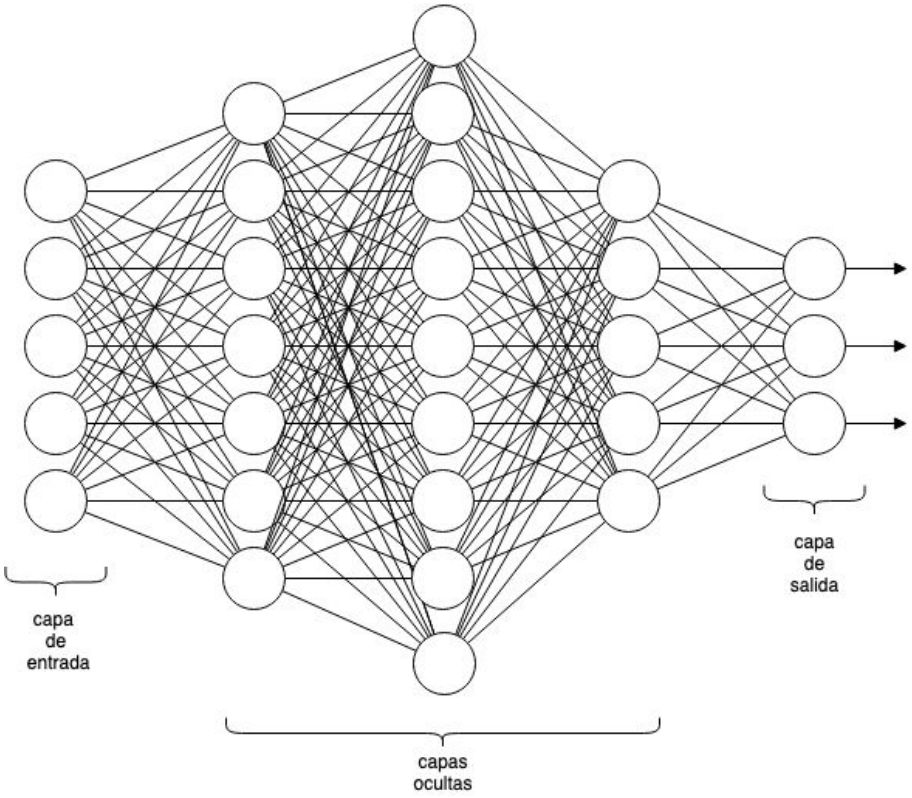

Figura 4: Red neuronal.

La capa de entrada simplemente pasa los valores de entrada a la próxima capa. Las capas siguientes a la capa de entrada son llamadas capas ocultas por no estar directamente expuestas a los valores de entrada. A la última capa se le llama 'capa de salida' y es la responsable de devolver el, o los, resultados calculados por la red. A las redes organizadas de esta forma, con una capa de entrada, una o más capas ocultas y una de salida se las conoce también como perceptrón multicapa (MLP, por su sigla en inglés). Cuando todas las neuronas de una capa están conectadas a cada neurona de la capa anterior se le llama capa densamente conectada (Dense).

Una red profunda puede verse como una operación compuesta por múltiples etapas distintas de procesamiento, donde la información se purifica a medida que es procesada a su paso por sucesivos filtros. A esta forma de aprendizaje se le llama aprendizaje profundo (Deep Learning).

Función de pérdida - Para poder controlar la salida de una red neuronal, primero es necesario poder medir la diferencia entre el valor esperado y el valor calculado por la red. Este es el trabajo de la función de pérdida. La función de pérdida toma las predicciones de la red y los valores verdaderos de la variable, luego calcula la diferencia entre sí, midiendo de esta forma el comportamiento de la red para este ejemplo específico. Este valor calculado es entonces utilizado

como realimentación para ajustar los pesos, de forma que se reduzcan las diferencias. Dicho ajuste es el trabajo del algoritmo de optimización. En el presente Trabajo Final Integrador la función de pérdida que se utiliza es el error absoluto medio.

Algoritmo de optimización El algoritmo de optimización se utiliza para minimizar o maximizar una función objetivo. Determina la actualización de la red neuronal, minimizando la función de pérdida. Su implementación se basa en la técnica de optimización 'Descenso del Gradiente'. En el presente Trabajo Final Integrador el algoritmo de optimización que se utiliza para los modelos de aprendizaje automático es RMSProp (Root Mean Square Propagation) [38]. RMSProp es una modificación del algoritmo 'Descenso del Gradiente' que utiliza diferentes tasas de aprendizaje para las variables, teniendo en cuenta únicamente los gradientes más recientes. Divide la tasa de aprendizaje para un determinado peso por un promedio calculado de las magnitudes de los gradientes más recientes de ese peso.

Entrenamiento Para el entrenamiento de una red neuronal mínimamente se requiere de las capas de la red, los datos de entrada, los objetivos de salida, la función de pérdida y el algoritmo de optimización. En este esquema, la red compuesta por capas conectadas, mapea los datos de entrada a las predicciones obtenidas. Luego, la función de pérdida compara dichas predicciones con los objetivos de salida esperados, obteniendo así una medida del error. El algoritmo de optimización utiliza dicha medida para actualizar los pesos sinápticos de la red neuronal. Inicialmente a los pesos de la red se le asignan valores aleatorios, pero a partir de cada muestra de datos que la red procesa la función de pérdida se minimiza, luego la diferencia entre los valores de salida esperados y calculados se minimiza, y los pesos sinápticos se ajustan. Es decir, que los pesos de la red se actualizan a partir de los errores calculados para cada muestra. Esto representa en términos generales el proceso de entrenamiento. El resultado final, luego de suficientes iteraciones es un modelo que representa o generaliza los datos involucrados.

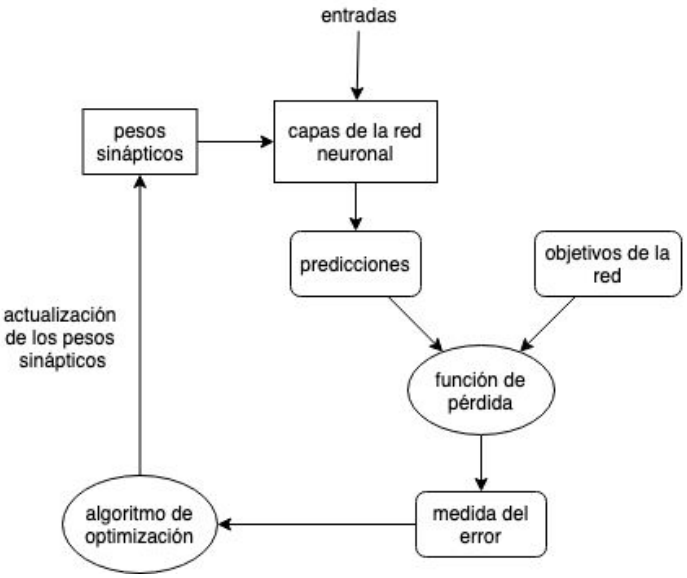

Figura 5: Entrenamiento de una red neuronal.

Datos de entrenamiento y testeo Antes de iniciar el proceso de aprendizaje es necesario particionar los datos. Éstos se dividen en dos conjuntos: un conjunto para entrenamiento y un conjunto para testeo. El conjunto para entrenamiento es el más grande y contiene los datos que serán utilizados para la creación de los modelos de aprendizaje automático y el estadístico. Luego, el conjunto para testeo es utilizado para la evaluación de los modelos creados.

3.2.2. Red neuronal recurrente

Una red neuronal recurrente (RNN, por su sigla en inglés) procesa secuencias por medio de la iteración de las secuencias de elementos, manteniendo un estado con información relativa a lo procesado hasta el momento. Una red neuronal recurrente es una red neuronal que contiene ciclos internos que realimentan la red, generando así memoria. Dicha memoria le permite a la red aprender y generalizar a lo largo de secuencias de entradas en lugar de patrones individuales. El estado de la red neuronal recurrente es reestablecido cada vez que procesan dos secuencias diferentes e independientes, por lo que se considera a una secuencia como un dato singular, una sola entrada en la red. Dicho dato ya no es procesado en un solo paso, sino que la red internamente cicla sobre la secuencia de elementos.

En la figura 6 se muestra el comportamiento de una red neuronal recurrente compuesta solamente por una neurona. En cada paso temporal t, la neurona recurrente recibe las entradas x_t y también su propia salida producida por el paso temporal previo y_{t-1}.

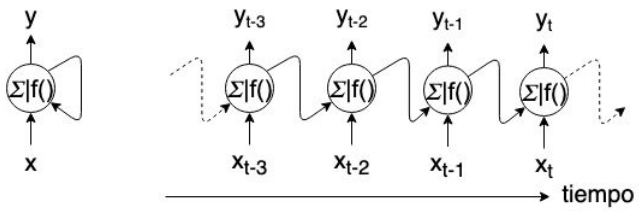

Figura 6: Neurona recurrente (izq.) - Neurona recurrente desplegada en el tiempo (der.)

Cada neurona recurrente tiene dos grupos de pesos sinápticos, uno para las entradas x_t y otro para las salidas del paso temporal previo y_{t-1}.

Las redes neuronales recurrentes tienen un problema, a pesar de que en teoría deberían retener información en un tiempo 't' sobre entradas procesadas varios pasos temporales atrás, en la práctica, dichas dependencias a largo plazo resultan imposibles de aprender. A este problema se lo conoce como 'Desvanecimiento del Gradiente'. La red eventualmente pierde su capacidad de ser entrenada a medida que se le agregan capas. Las razones teóricas de tal efecto fueron estudiadas en [30, 31].

3.2.3. Red neuronal con memoria a corto plazo extendida

La red neuronal con memoria a corto plazo extendida (LSTM, por su sigla en inglés) le agrega una variante a la red neuronal recurrente básica. Agrega una manera de transportar información a lo largo de varios pasos temporales y almacena información para su posterior uso, previniendo de esta manera que las señales más viejas se desvanescan gradualmente durante el procesamiento.

El objetivo de esta capa es darle solución al problema mencionado anteriormente como 'Desvanecimiento del Gradiente' de las redes neuronales recurrentes. Para esto, la capa LSTM extiende la arquitectura de la RNN adicionando una célula de memoria y una serie de compuertas que manipulan el flujo de información a través de la red. Hay tres tipos de compuertas trabajando en forma colaborativa:

- La compuerta de *entrada* toma decisiones sobre la actualización de la información entrante.
- La compuerta de *olvido* decide si la información debe ser olvidada o mantenida.
- La compuerta de *salida*, a partir del procesamiento de los resultados de las compuertas de *entrada* y *olvido* devuelve el cálculo final de la célula.

3.2.4. Red neuronal convolucional

Las redes neuronales convolucionales son un tipo de red neuronal diseñado originalmente para procesar datos de imágenes, pero las mismas propiedades que hacen propicias a las redes neuronales convolucionales para problemas de visión

de computadoras las hacen muy relevantes para procesar señales. El tiempo puede ser tratado como una dimensión espacial, como la altura o el ancho de una imagen 2D. Estas son las redes convolucionales 1D y pueden ser competitivas con las redes neuronales recurrentes para resolver problemas de procesamiento secuencial, usualmente a un costo computacional menor.

Las capas de convolución 1D obtienen nuevas secuencias convolucionadas a través de filtros que interpretan ciertas características de las secuencias originales que permiten reconocer patrones locales en la misma.

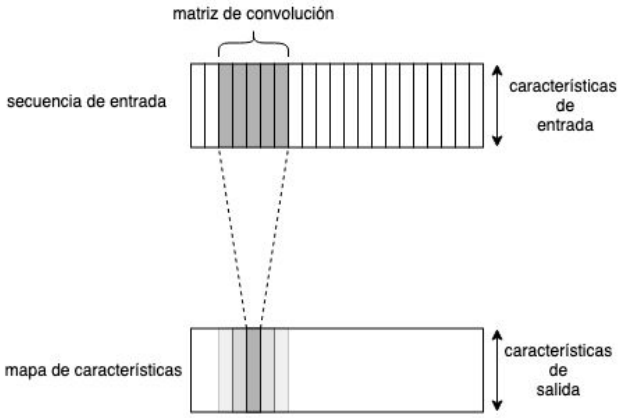

Figura 7: Convolución 1D

En una red neuronal convolucional hay tres tipos de capas, las capas convolucionales, de pooling y densamente conectadas.

La capa convolucional es la capa más importante de una red neuronal convolucional. Las neuronas de esta capa no están conectadas a cada uno de los valores de entrada, solo a los valores dentro de sus campos receptivos. Luego, en la siguiente capa convolucional, cada neurona está conectada solamente a neuronas ubicadas dentro de un sector reducido de la capa anterior. La diferencia fundamental entre una capa densamente conectada y una convolucional es que las capas densas aprenden patrones globales de sus entradas, mientras que las convolucionales aprenden filtros (o Kernels) que modifican la señal original, generando descriptores o mapas de características. Esta arquitectura le permite a la red concentrarse en características más simples en las primeras capas y agruparlas luego en características más complejas en las capas siguientes. Las capas convolucionales se componen de estructuras, filtros y mapas de características. Los filtros son esencialmente los distintos grupos de pesos sinápticos de las neuronas de la capa. La entrada de cada neurona está conformada por un grupo de valores contiguos que componen el campo receptivo. Durante el entrenamiento, la red neuronal convolucional encuentra los filtros más útiles para su tarea y aprende a combinarlos en patrones más complejos. El mapa de características es la salida o resultado de una capa utilizando un determinado filtro. La información que cada uno contiene resalta las secciones en la secuencia más similares al filtro.

La capa de pooling tiene como objetivo reducir la muestra anteriormente procesada, lo que disminuye la carga computacional, utilización de memoria y número de parámetros, extrayendo subsecuencias de una entrada y devolviendo el valor máximo o promedio.

En una arquitectura típica de una red neuronal convolucional se apilan un grupo de capas convolucionales, luego una capa de pooling. Dicha estructura mencionada normalmente se repite. La entrada se reduce más y más a su paso por la red, a la vez que aumenta la cantidad de mapas de características. Al final de la pila una red neuronal densamente conectada es agregada.

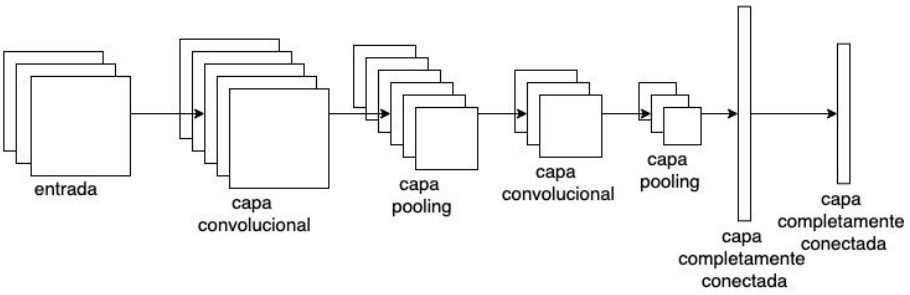

Figura 8: Arquitectura típica de una red nueronal convolucional.

3.2.5. Ensamble

Una poderosa técnica para la obtención buenos resultados es el ensamble de modelos [26]. El ensamble consiste en la reunión o agrupación de las predicciones de un conjunto de modelos diferentes para la producción de mejores predicciones. Para el ensamble se presupone que modelos buenos, diferentes entre sí y entrenados en forma independiente, probablemente sean buenos por razones diferentes. Cada modelo capta diferentes aspectos de los datos para realizar predicciones. En esencia cada modelo trata de modelar los datos desde su propia perspectiva. Luego agrupando dichas perspectivas correctamente es posible obtener una descripción mucho más precisa de los datos.

Una manera simple de agrupar las predicciones de un conjunto de modelos es realizar un promedio de las mismas. Se prioriza en forma equitativa a cada modelo. La desventaja de dicha técnica es que la precisión de la predicción final puede no ser tan buena como la del mejor predictor del grupo.

Una manera de ensamblar modelos que entrega mejores resultados es realizando un promedio ponderado, aprendiendo los pesos a partir de los datos para entrenamiento y de las predicciones de cada modelo. A los modelos más precisos se les otorga un mayor peso, mientras que a los modelos menos precisos se les otorga un menor peso. Luego, para encontrar el conjunto de pesajes más adecuado para el ensamble se utiliza una algoritmo de optimización.

Son muchas las variantes posibles, pero la clave para que el ensamble funcione efectivamente reside en la diversidad del conjunto de modelos. Si todos

los modelos se encuentran sesgados en un mismo sentido, entonces el ensamble retendrá dicho sesgo. Si los modelos están sesgados en direcciones diferentes, los sesgos se cancelarán entre sí, y el ensamble será más robusto y preciso. Por esta razón es recomendable ensamblar modelos tan precisos y diferentes como sea posible. Esto típicamente significa utilizar modelos de arquitecturas diferentes.

3.3. Redes neuronales aplicadas a series temporales

A continuación se mencionan ejemplos actuales de la literatura donde se utilizan modelos basados en redes neuronales para la predicción de series temporales.

En el artículo 'Statistical and Machine Learning forecasting methods: Concerns and ways forward' [13] se utiliza un conjunto de 1045 series temporales para calcular la performance de ocho métodos estadísticos y ocho métodos de aprendizaje automático. Los modelos se desarrollan utilizando las primeras $n - 18$ observaciones, donde n es la longitud de la serie. Luego se calculan 18 predicciones a futuro y su precisión es evaluada y comparada. Entre los métodos de aprendizaje automático que se utilizan se encuentran modelos creados a partir de redes neuronales basados en capas MLP y LSTM. En la arquitectura del modelo MLP, la red neuronal cuenta con una sola capa oculta. Se definen la cantidad N de nodos de entrada entre un rango de valores de 1 a 5 mediante un proceso de validación, siendo las entradas la últimas observaciones previas al objetivo de la predicción, mientras que el número de nodos de la capa oculta se establece en $2N + 1$. En dicho estudio, el modelo utilizado para implementar la red neuronal recurrente se compone de una capa oculta que contiene los nodos recurrentes, y una de salida que contiene uno o más nodos lineales. La capa de entrada consiste de tres nodos de entrada, seis unidades LSTM que forman la capa oculta y un nodo lineal en la capa de salida para todas las series del conjunto de datos. Las conclusiones que se obtienen de las comparaciones muestran resultados satisfactorios de cada método y apuntan a explicar el motivo de la superioridad de los estadísticos.

En el artículo 'Fast ES-RNN: A GPU Implementation of the ES-RNN Algorithm' [16] se describe el algoritmo que entregó los mejores resultados de la competencia M4 [15]. Los datos utilizados en esta competencia se componen de 100.000 series temporales con frecuencia anual, trimestral, mensual, semanal, diaria y horaria de origen financiero, industrial, económico, demográfico y climático. Es un modelo híbrido denominado ES-RNN (Exponential Smoothing-Recurrent Neural Network), que utiliza el método estadístico Exponential Smoothing (Suavizado Exponencial) [40] en combinación con redes neuronales. El algoritmo se divide en dos secciones distintivas. Una sección de pre procesamiento que utiliza el método estadístico y una sección de redes neuronales LSTM que actualiza los parámetros de la capa anterior. La sección de redes neuronales utiliza cuatro capas ocultas LSTM con conexiones especiales entre sí. Al final de la arquitectura se encuentra una simple capa lineal que adapta la salida de la red neuronal recurrente con la predicción final.

En el estudio 'An ensemble of neural networks for weather forecasting' [22] se presenta la aplicabilidad de las redes neuronales artificiales para la predicción

de series temporales con información climática. Los datos incluyen temperatura, velocidad del viento y humedad relativa. Se propone un método de ensamble basado en redes neuronales con capas MLP y LSTM, en el que sus pesos son determinados en forma dinámica a partir de las respectivas certezas de los resultados de las redes. Cuando más certera una red parece ser de su decisión, más alto es el peso que se le asigna. De acuerdo a lo expuesto, las arquitecturas de las redes se obtienen a partir de un proceso de prueba y error. Las redes MLP y LSTM contienen únicamente una capa oculta compuesta por 72 neuronas. Se obtiene dicha cantidad luego de analizar resultados con 24, 48, 72 y 120 neuronas en la capa oculta. Se observa que, en este caso, con el agregado de más capas ocultas se perjudica la precisión de los resultados. Los experimentos apuntan principalmente a probar que es posible obtener resultados con una precisión satisfactoria, y a su vez, a comparar el desempeño del método combinado con los métodos individuales.

4. Marco experimental

4.1. Origen de los datos

En el presente Trabajo Final Integrador se utilizó un conjunto de datos con información climática que fué grabada con el objetivo de ser analizada en la estación climática en el Instituto de Biogeoquimica Max Planck en Jena, Alemania [www.bgc-jena.mpg.de/wetter]. El mismo contiene en total 14 variables diferentes (tal como temperatura del aire, presión atmosférica, humedad, dirección del viento, etc) grabadas cada 10 minutos, a lo largo de varios años.

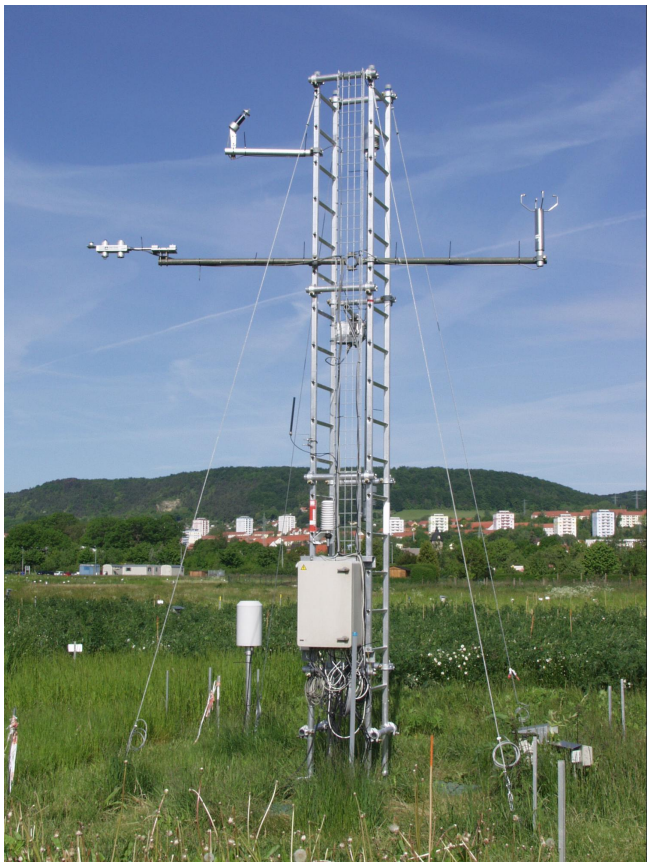

Figura 9: De acuerdo a la documentación disponible de la Estación Climática [29] casi todos los instrumentos están sujetos a un mástil de 10 metros de altura ubicado al descubierto. Este mástil es también utilizado para llevar temporalmente otros instrumentos para comparaciones o calibraciones.

En el cuadro 1 se listan todas las variables medidas con su correspondiente instrumento y la compañía que los produce.

Los sistemas de adquisición de datos, fuentes de energía, laptops y dispositivos similares se ubican dentro de lo que llaman casillero modular, de la compañía

Cuadro 1: Mediciones

Variable	Instrumento	Productor
Temperatura del Aire	KPK1/5-ME	Mela Sensortechnik
Humedad Relativa del Aire	KPK1/5-ME	Mela Sensortechnik
Presión de Aire	PTB101B	Vaisala
Velocidad del Viento	A100R	Vector Instruments
Dirección del Viento	W200P	Vector Instruments
Radiación entrante de Onda Corta	CM11	Kipp & Zonen
Radiación Activa Fotosintética	PAR Lite	Kipp & Zonen
Cantidad de Precipitación	5.4032.35.008	Thies
Precipitación si/no	5.4105.00.000	Thies
Concentración de CO_2	LI6262	Licor

Rittal, a un costado de la base del mástil.

Para la adquisición de las señales de los instrumentos mencionados se utiliza un dispositivo denominado datalogger CR10X (Campbell Scientific). Dicho dispositivo realiza una lectura de sensores cada 10 segundos. Promedia totales y máximos de las variables sobre períodos de 10 minutos, internamente calculados y almacenados en la memoria. El programa LoggerNet (Campbell Scientific), se ejecuta en una computadora separada, dentro de la red local del instituto, que realiza una conexión cada 10 minutos al datalogger para descargar los últimos datos o todos que no han sido descargados.

Todos los datos, desde el comienzo de su recolección se encuentran disponibles gratuitamente y pueden descargarse desde la página web como archivos comprimidos. [www.bgc-jena.mpg.de/wetter/weather_data.html]

En el cuadro 2 se listan todas las variables obtenidas y calculadas disponibles con sus respectivas unidades de medida junto con una descripción.

Cuadro 2: Variables obtenidas

Símbolo	Unidad	Variable
Date Time	DD.MM.YYYY HH:MM (MEZ)	Fecha y hora del registro de datos
p	mbar	Presión del aire
T	C^o	Temperatura del aire
Tpot	K	Temperatura potencial
Tdew	C^o	Temperatura de rocío
rh	%	Humedad relativa
VPmax	mbar	Saturación de presión de vapor de agua
VPact	mbar	Presión de vapor de agua real
VPdef	mbar	Déficit de presión de vapor de agua
sh	gkg^{-1}	Humedad específica
H2OC	$mmolmol^{-1}$	Concentración de vapor de agua
rho	gm^{-3}	Densidad del aire
wv	ms^{-1}	Velocidad del viento
max. wv	ms^{-1}	Velocidad máxima del viento
wd	^{o}	Dirección del viento

4.2. Hardware y software utilizado

Los principales componentes de Hardware utilizados para realizar los experimentos son:

- Procesador Intel Core i7-8700 3.20GHz
- Memoria del sistema 32GB DDR4 2666MHz

Por otra parte, las versiones de los principales componentes de Software (ver Anexo) utilizados para realizar los experimentos son:

- Ubuntu 18.04.3 LTS
- Keras 2.3.1
- Tensorflow 2.0.0
- Numpy 1.17.4
- Pandas 0.25.3
- Matplotlib 3.1.1
- Statsmodels 0.10.1
- Scipy 1.3.2
- Python 3.6.9
- Anaconda Navigator 1.9.7
- Jupyter Notebook Server 6.0.0

5. Desarrollo

El desarrollo de los experimentos no se lleva a cabo con la totalidad de datos disponibles. Originalmente se realizaron pruebas con el conjunto de datos completo, pero los altos requerimientos de hardware, en particular de memoria RAM (Random Access Memory) al trabajar con los métodos de aprendizaje automático resultaron limitantes. Esto se debe a la forma en la que es necesario preparar los datos para su presentación en las redes neuronales. Las estructuras de datos crecen considerablemente en función del tamaño de ciertos parámetros de diseño, como la cantidad de observaciones pasadas de cada muestra, lo cual obliga a tener siempre en cuenta la cantidad de datos que se están manejando y la eficiencia de las estructuras de datos con las que se los manipula y procesa.

Para operaciones como la mezcla aleatoria de muestras (que solo se realiza para los modelos basados en redes neuronales) se utilizan estructuras de datos y operaciones de la librería TensorFlow (ver Anexo). Dicha librería demostró ser muy eficiente para esta tarea, más aún que librería NumPy para las mismas operaciones de mezcla.

Se toma en primer instancia un subconjunto de datos desde la fecha y hora '2009-01-01 00:10:00' hasta la fecha y hora '2011-12-31 23:50:00'. Luego, de las 15 características disponibles se seleccionan 5 teniendo en cuenta factores como su rango y sus patrones estacionales, con el objetivo de que el subconjunto elegido sea heterogéneo, y a su vez, representativo del original. Las variables seleccionadas son las siguientes:

- $p(mbar)$ - Presión del aire
- $T(C^o)$ - Temperatura del aire
- $rh(\%)$ - Humedad relativa
- $rho(g/m_3)$ - Densidad del aire
- $wv(m/s)$ - Velocidad del viento

5.1. Análisis general de los datos

Para el análisis exploratorio preliminar y general de los datos, de utilidad para todos los métodos predictivos, se realizan las siguientes operaciones:

- Verificación de valores faltantes. Si los valores de una variable no son capturados durante un intervalo de tiempo es necesario tomar alguna medida, por ejemplo reemplazar los valores faltantes por un promedio de los vecinos más cercanos. En el caso particular del conjunto de datos utilizado no hay inconvenientes de este tipo.
- Inspección en forma aleatoria de los valores de cada variable. Se verifica que exista una coherencia secuencial, teniendo en cuenta la periodicidad con la que se realizan las mediciones.

- Un gráfico de cada variable. Esto permite ver en forma global si los datos son consistentes y uniformemente medidos o si hay cambios que sugieren modificaciones en el método de medición. De esta forma también se hacen evidentes los rangos de los valores

A continuación se muestran los gráficos de las variables que se utilizan. Los datos que se utilizan para entrenamiento están en color azul, y en color naranja los que se utilizan para testeo.

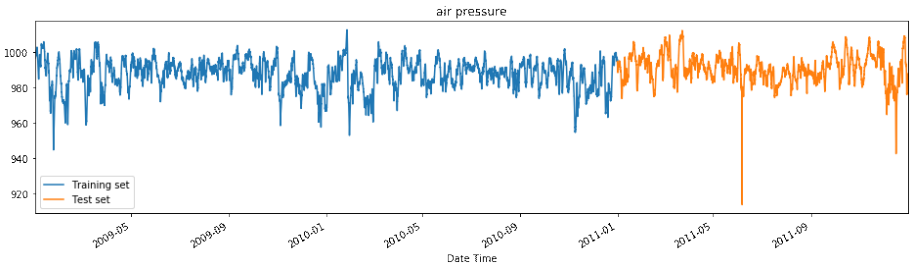

Figura 10: Variable con información de la presión del aire $p(mbar)$.

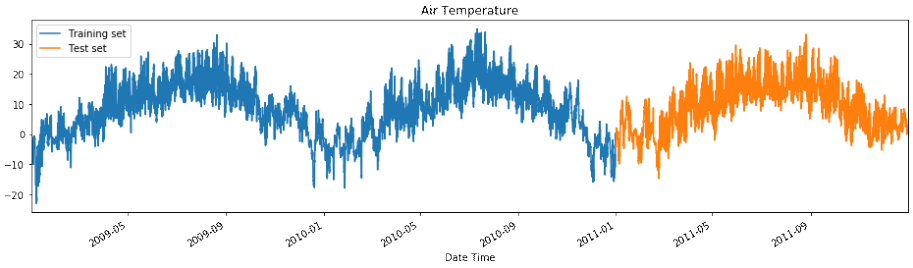

Figura 11: Variable con información de la temperatura del aire $T(C^o)$

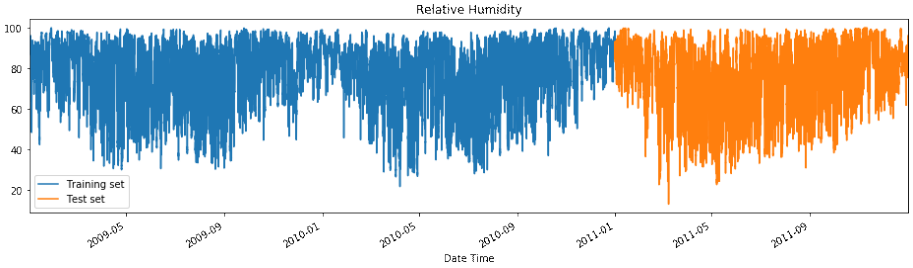

Figura 12: Variable con información de la humedad relativa $rh(\,\%)$

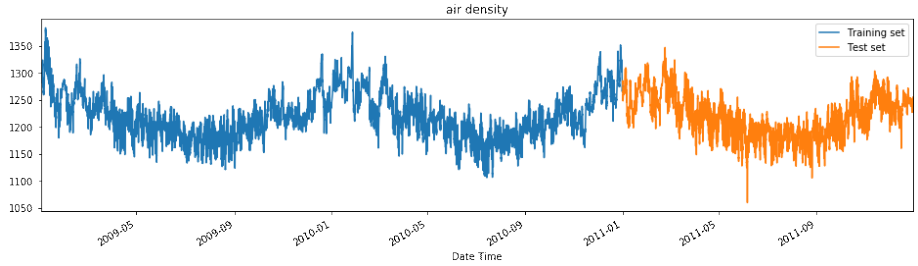

Figura 13: Variable con información de la densidad del aire $rho(g/m_3)$

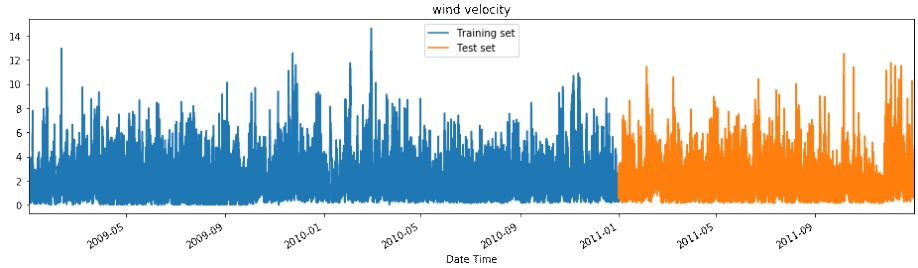

Figura 14: Variable con información de la velocidad del viento $wv(m/s)$

5.2. Modelo estadístico - Experimentos

El procedimiento para la creación del modelo VAR involucra los siguientes pasos:

1. Se analiza por separado cada una de las variables de la serie.
2. Se realiza una división de los datos en un grupo para entrenamiento y otro grupo para testeo.
3. Se verifica la estacionariedad de los datos.
4. Se aplican las transformaciones necesarias para hacer los datos de entrenamiento estacionarios.
5. A partir de los coeficientes y modelos de transformación obtenidos en el paso anterior se aplican las transformaciones necesarias para hacer estacionaria a la serie completa.
6. Se realiza una nueva división de los datos transformados en un grupo para entrenamiento y otro grupo para testeo.
7. Se determina el orden óptimo (p) del modelo VAR.
8. A partir de los datos de entrenamiento transformados de ajusta el modelo VAR.
9. Se suministra al modelo los últimos (p) valores observados para generar una predicción.

10. Se revierte la transformación de la predicción al dominio de la serie original.

11. Se evalúa el modelo utilizando los datos de testeo originales.

A continuación se realiza una explicación de cada uno de los pasos mencionados.

5.2.1. Análisis y preparación de los datos

Luego del análisis preliminar de los datos se realiza una separación de los datos en dos grupos. Uno para entrenamiento con fecha y hora desde '2009-01-01 00:00:00' hasta '2010-12-31 23:50:00', y otro para testeo desde '2011-01-01 00:00:00' hasta '2011-12-31 23:50:00'. En cuanto a cantidades, los datos para testeo representan una tercera parte del total. Como el modelo VAR requiere que la serie temporal completa sea estacionaria, es necesario realizar un análisis sobre cada una de sus variables. Se realiza una inspección visual y verificación numérica de estacionariedad sobre cada una de las subseries o variables que la componen.

Con respecto a la inspección visual de las subseries, todas presentan un marcado patrón estacional anual y diario. Esto se puede apreciar claramente en los gráficos de autocorrelación. En el presente Trabajo Final Integrador se utiliza la función plot_acf() de la librería Statsmodels (ver Anexo) para graficar el ACF de una serie.

En las figuras 15 y 16 se muestran respectivamente las funciones de autocorrelación de los primeros 200 y 1000 valores de la variable correspondiente a la temperatura de la serie temporal. Representan los gráficos de la cuantificación, en un rango de valores entre 1 y -1, de la autocorrelación de las observaciones respectivamente mencionadas. Teniendo en cuenta que el intervalo temporal entre observaciones es de 10 min, el patrón cíclico que se observa cada 144 observaciones corresponde al transcurso de 24 horas.

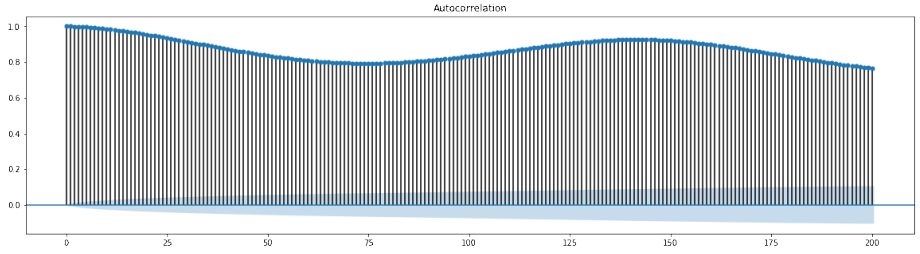

Figura 15: Función de autocorrelación de las primeras 200 observaciones de la variable de temperatura $T(C^o)$ graficada mediante $plot\_acf(datos\_temperatura,\ lags = 200)$.

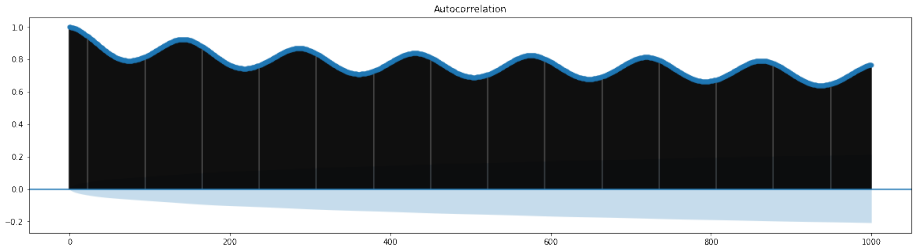

Figura 16: Función de autocorrelación de las primeras 1000 observaciones de la variable de temperatura $T(C^o)$ graficada mediante $plot\_acf(datos\_temperatura,\ lags = 1000)$.

En la figura 17 se muestra el gráfico de autocorrelación de los primeros 100000 valores de la variable correspondiente a la temperatura de la serie temporal. El patrón cíclico que se observa es aproximadamente cada 52000 observaciones, es decir, corresponde al transcurso de un año.

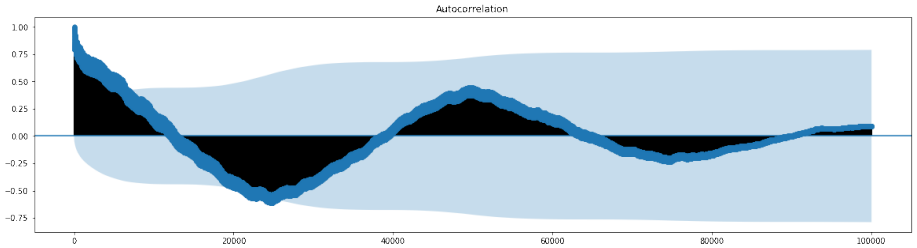

Figura 17: Función de autocorrelación de las primeras 100.000 observaciones de la variable de temperatura $T(C^o)$ graficada mediante $plot\_acf(datos\_temperatura,\ lags = 100000)$.

Para la confirmación numérica de estacionariedad de la serie se utiliza el test KPSS sobre cada variable. El mismo arroja para cada variable un valor de probabilidad de 0.010000 rechazando la estacionariedad. Luego, para hacer estacionarios los datos se aplican las siguientes transformaciones:

1. Transformación Box Cox. Esta transformación potencial es utilizada para estabilizar la varianza a lo largo del tiempo. Comprime el espacio entre los valores más grandes, pero no entre los más chicos. De esta forma se reduce el efecto de los valores atípicos y se aproxima la distribución de los datos a una distribución normal.

2. Diferenciación. Se aplica a la serie completa una diferenciación de primer orden. De esta forma se reducen las tendencias y los patrones estacionales.

3. Estandarización. Esta transformación tiene el efecto de centrar los datos. De esta forma, la media de los datos se convierte en 0, y su desviación estándar en 1.

4. Normalización. Esta transformación modifica la escala de los datos a un nuevo rango entre 0 y 1.

Las transformaciones se aplican primero sobre los datos para entrenamiento. A partir de dichas transformaciones se obtienen los parámetros lambda (λ) de la transformación Box Cox y los modelos de normalización y estandarización. Luego se aplican las mismas transformaciones a la serie original completa (entrenamiento + testeo) utilizando los parámetros y modelos de transformación obtenidos y ajustados anteriormente. De esta manera, los datos para testeo son transformados de acuerdo a los datos conocidos, para entrenamiento. Esta última serie completa transformada se utiliza luego para revertir las transformaciones de las predicciones del modelo predictivo.

Luego de las transformaciones se verifica que el test KPSS confirme la estacionariedad de cada variable o subserie del sistema. El mismo arroja para cada variable un valor de probabilidad de 0.100000 confirmando la estacionariedad. La serie completa, esta vez transformada, se divide nuevamente en un conjunto para entrenamiento y un conjunto para testeo. De esta forma ambos grupos pertenecen al mismo dominio de transformación, al igual que el modelo que se entrena, e inclusive las predicciones que se obtienen del mismo.

El modelo predictivo realiza predicciones en el dominio transformado. Luego para cambiar el dominio de dichas predicciones al de la serie original, es necesario revertir las transformaciones en orden inverso al aplicado originalmente. Para revertir una transformación de diferenciación se necesita de la serie completa diferenciada. Es necesario concatenar la predicción con los datos temporalmente anteriores de la serie, en el mismo dominio transformado (los cuales son datos conocidos). Luego, a dicha serie resultante se le aplica la transformación que revierte la diferenciación.

En las figuras 18, 19, 20, 21 y 22 se muestran los gráficos de los datos para entrenamiento de cada variable luego de las transformaciones, utilizadas como entrada del modelo Vector Autorregresivo.

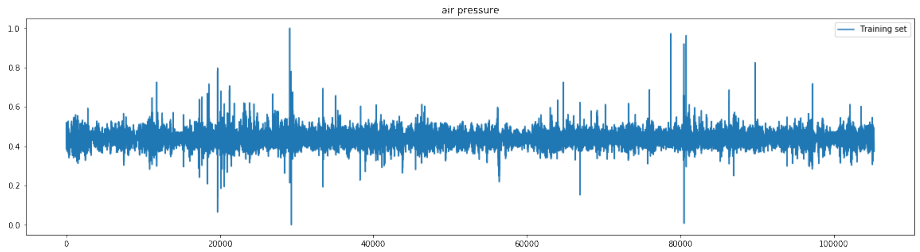

Figura 18: Variable con información estacionaria de la presión del aire $p(mbar)$

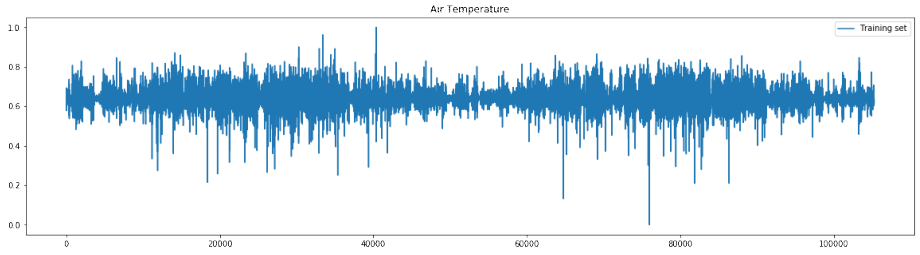

Figura 19: Variable con información estacionaria de la temperatura del aire $T(C^o)$

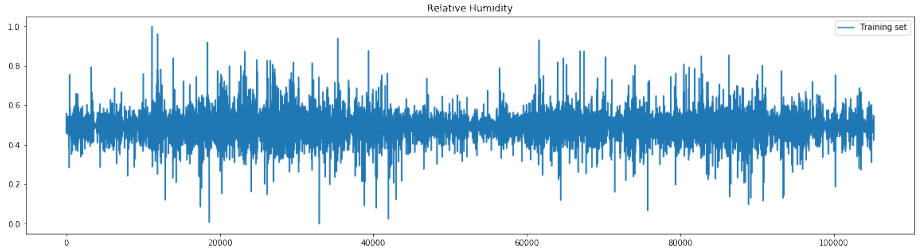

Figura 20: Variable con información estacionaria de la humedad relativa $rh(\,\%)$

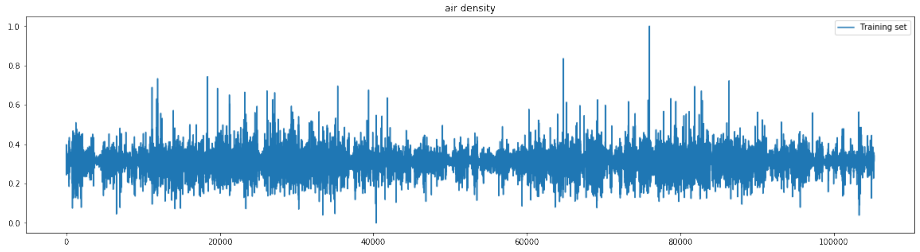

Figura 21: Variable con información estacionaria de la densidad del aire $rho(g/m_3)$

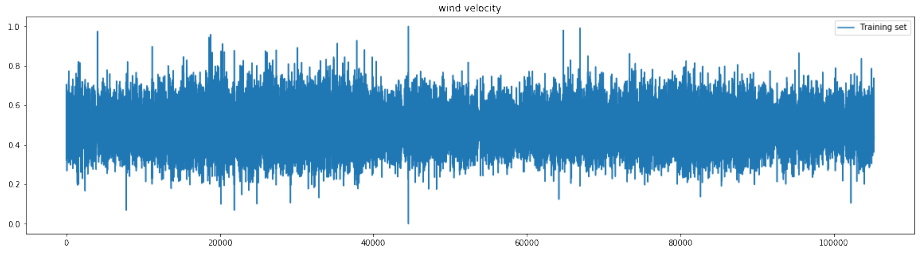

Figura 22: Variable con información estacionaria de la velocidad del viento $wv(m/s)$

5.2.2. Creación del modelo

A continuación se muestran los pasos para la creación del modelo estadístico:

$modelo = VAR(serie\_temporal\_estacionaria)$
$modelo.select\_order().summary()$
$modelo.fit(orden)$

Se crea el modelo utilizando la clase VAR, de la librería Statsmodels (ver Anexo), la cuál recibe como parámetro la serie temporal estacionaria. Luego se define su orden de retraso (p), el número de pasos temporales u observaciones pasadas con los que el modelo determina su sistema de ecuaciones. Para encontrar el orden de retraso más apropiado se utiliza la función $select\_order()$, accesible a través de la clase VAR, la cuál evalúa iterativamente distintos órdenes de retraso sobre el modelo dado y retorna por medio de un resúmen el número óptimo de acuerdo a cada criterio de información. Los valos obtenidos en el caso del presente Trabajo Final Integerador se muestran en el cuadro 3.

Cuadro 3: Resúmen

Criterio	Orden Óptimo
AIC	164
BIC	12
HQC	73

Para el ajuste del modelo se utiliza la función $fit()$, también accesible a través de la clase VAR, y un orden retraso igual a 164 el cuál es devuelto como óptimo según el criterio de información AIC.

5.2.3. Evaluación del modelo

Una vez que se dispone del modelo ajustado a un orden (p), para realizar una predicción es necesario alimentar el modelo con las últimas (p) observaciones de datos pasados.

Se realizan predicciones de las 5 variables en simultáneo con la finalidad de obtener mediciones de las predicciones en los siguientes horizontes predictivos:

- 10 minutos, es decir, la predicción de la próxima observación de cada variable.
- 24 horas, es decir, la predicción de las próximas 144 observaciones de cada variable.
- 48 horas, es decir, la predicción de las próximas 288 observaciones de cada variable.
- 72 horas, es decir, la predicción de las próximas 432 observaciones de cada variable.

Como el modelo es entrenado con datos transformados, su dominio predictivo también está transformado. Esto significa que luego de realizar las predicciones, es necesario cambiar su dominio, revertir su transformación implícita. Para esto son necesarios los parámetros, modelos y series obtenidos durante el proceso de transformación de los datos originales. Como se explicó anteriormente, es necesario revertir las transformaciones en orden inverso al aplicado originalmente.

En el caso de los modelos de aprendizaje automático, se entrenaron modelos distintos para cada horizonte. En cambio en el caso del modelo VAR el modelo es el mismo, es decir, todas las mediciones se obtienen a partir de un único predictor.

5.3. Modelos de aprendizaje automático - Experimentos

El procedimiento para la creación de los modelos de aprendizaje automático involucra los pasos que se explican a continuación.

5.3.1. Preparación de los datos

Luego del análisis preliminar de los datos se realiza una separación de los mismos en dos grupos, al igual que para el modelo estadístico. Uno para entrenamiento y otro para testeo con los mismos rangos de valores.

Se obtienen la media y la desviación estándar del conjunto de datos para entrenamiento, se sustrae el valor de la media obtenido a ambos conjuntos y se divide, también a ambos conjuntos, por el valor de la desviación estándar obtenido. De esta forma la serie completa queda transformada en función de los datos para entrenamiento. En general no es una buena práctica alimentar una red neuronal con valores relativamente grandes o datos heterogéneos, con valores en rangos de magnitud diferentes, por ejemplo datos donde una variable pertenezca al rango 0-1 y otra pertenezca al rango 100-200. Hacer eso puede desencadenar actualizaciones de gradientes que prohiban a la red alcanzar la convergencia, perjudicando el aprendizaje.

Luego de las transformaciones, es necesario estructurar los datos como un problema de aprendizaje supervisado. En el aprendizaje supervisado, a cada conjunto de variables de entrada X le corresponde un conjunto de variables de salida Y, y un algoritmo es utilizado para aprender una función de mapeo desde la entrada a la salida.

$$Y = f(X)$$

El objetivo es aproximar el mapeo subyacente real tan bien, que para nuevos datos de entrada X, sea posible predecir las variables de salida Y correspondientes. Se conocen las respuestas correctas, el algoritmo realiza predicciones en forma iterativa sobre los datos para entrenamiento y es corregido realizando actualizaciones. El aprendizaje se detiene cuando el algoritmo alcanza un nivel de desempeño aceptable. Por ejemplo, a partir de una serie temporal de una sola variable representada por un vector de observaciones de la siguiente forma:

$$[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]$$

la serie temporal anterior puede ser expresada como un problema de aprendizaje supervisado con un conjunto de pasos temporales u observaciones pasadas temporalmente consecutivas como entrada y una observación como salida, de la siguiente forma:

X	Y
$[1, 2, 3]$	$[4]$
$[2, 3, 4]$	$[5]$
$[3, 4, 5]$	$[6]$
...	

Para el caso de las series multivariadas, como la serie utilizada en el presente Trabajo Integrador Final, los datos de las muestras utilizadas como entrada y las de salida, en principio, deben quedar estructurados tridimensionalmente. Las tres dimensiones de dicha entrada son:

- Muestras. Una secuencia es una muestra. Un lote se compone de una o más muestras. La cantidad de muestras de entrada y de salida es la misma. Esto significa que a cada muestra de entrada le corresponde una muestra de salida.
- Pasos temporales. Un paso temporal es un punto de observación en una muestra. Una muestra se compone de múltiples pasos temporales. La cantidad de pasos temporales de las muestras de salida es la cantidad de predicciones, o pasos temporales a futuro que el modelo aprende a predecir a partir de una muestra de entrada. La cantidad de pasos temporales de las muestras de entrada es la cantidad de observaciones de cada variable que el modelo utiliza para aprender a predecir la salida.
- Características (*features*). Una característica es una variable de una observación. Un paso temporal se compone de una o más características. Las características presentes en las muestras de salida son las que el modelo aprende a predecir a partir de una muestra de entrada.

La estructura de datos que representa un lote de muestras de entrada se resume comúnmente de la siguiente manera:

$$[\textit{muestras, pasos temporales de entrada, características de entrada}]$$

Similarmente, la estructura de datos que representa a un lote de muestras de salida se resume comúnmente de forma:

$$[\textit{muestras, pasos temporales de salida, características de salida}]$$

En esta instancia los datos se encuentran estructurados en muestras, donde a cada entrada le corresponde una salida. En este punto es necesario realizar una mezcla aleatoria del orden de las muestras manteniendo la misma relación de orden entre entradas y salidas. De esta forma se altera la temporalidad entre

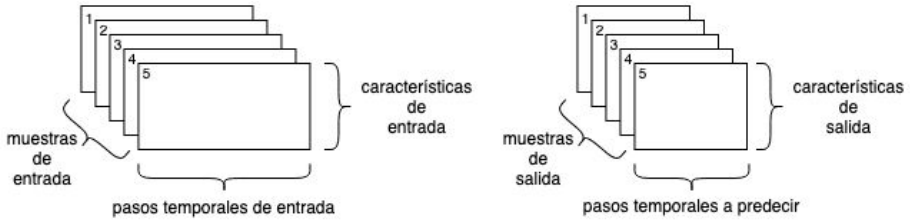

Figura 23: Estructura de las muestras

muestras. El objetivo es que las muestras que se le presenten a las redes neuronales sean, en lo posible, lo más diferentes entre sí. Esto favorece el proceso de actualización de pesos sinápticos de la red neuronal en el entrenamiento, mejorando el aprendizaje.

El último paso sobre la estructuración de los datos es el aplanado de las muestras de salida. Esto es necesario, ya que las salidas de las redes neuronales tienen únicamente dos dimensiones. La estructura final de las muestras de salida tienen el siguiente formato:

[*muestras, pasos temporales de salida * características de salida*]

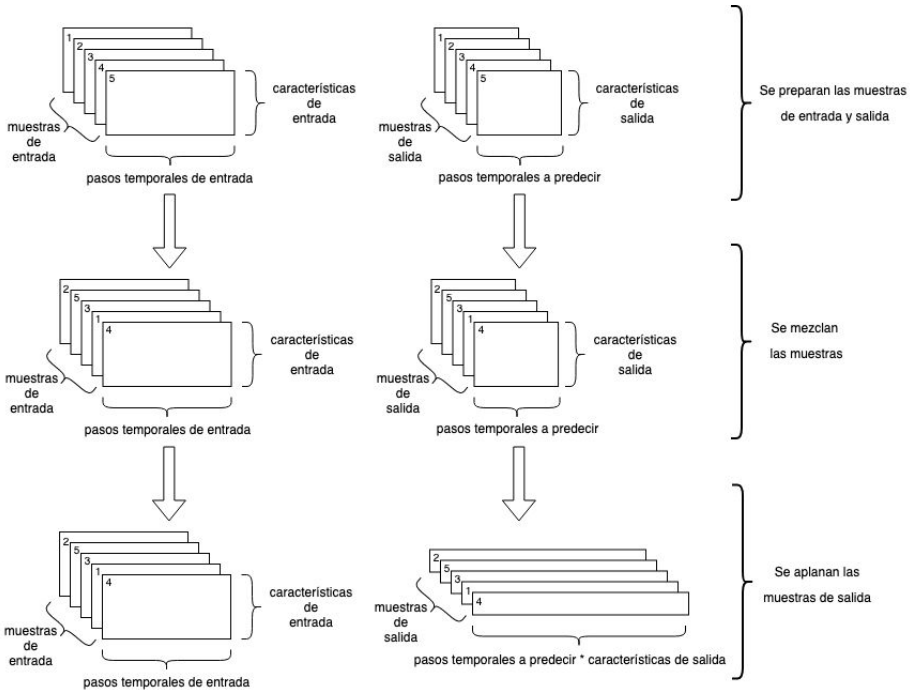

Figura 24: Proceso de transformación estructural de los datos

La cantidad de pasos temporales de las muestras de entrada en el presente trabajo se escogió mediante un proceso de prueba y error, mediante la verifica-

ción de los resultados obtenidos utilizando varios pasos temporales diferentes. Se utiliza finalmente un valor de 100 pasos temporales, pero vale mencionar que se observó que la influencia que ejerce el tamaño de dicha ventana de observaciones pasadas es diferente sobre cada variable. Como ejemplo de este caso, se observa que utilizar valores mayores a 100 mejora levemente los resultados obtenidos en las predicciones de la variable de presión de aire, pero empeora los de la variable de temperatura.

5.3.2. Predicciones de múltiples horizontes predictivos

Hay tres maneras alternativas para producir predicciones de períodos múltiples con modelos de aprendizaje automático [13].

- Predicción iterativa. Se genera una nueva predicción utilizando como entrada del modelo predictivo las predicciones realizadas anteriormente. Por ejemplo se genera a partir de los datos una primera predicción, luego se utiliza dicho resultado como punto de partida para realizar una segunda predicción. Dicho procedimiento se repite, hasta encontrar todas las predicciones necesarias. La ventaja de este enfoque es su simplicidad y reducida carga computacional. La desventaja es que a medida que el horizonte predictivo se incrementa, la precisión de las nuevas predicciones tiende a deteriorarse.
- Predicción directa. El enfoque directo produce múltiples predicciones a futuro en forma simultánea, en lugar de una a la vez, entrenando y aprovechando una red neuronal con salida de múltiples nodos, uno por cada horizonte predictivo. Por ejemplo, para la predicción de dos observaciones el primer nodo de salidas es responsable por la predicción temporalmente más cercana, y el segundo nodo de salida genera la predicción siguiente. Este enfoque es más complejo y computacionalmente más demandante que el iterativo.
- Predicción con múltiples redes neuronales. En este enfoque, son entrenadas múltiples redes neuronales de salida simple para la producción de las distintas predicciones. En lugar de entrenar una red neuronal para predecir en forma simultánea, varias redes neuronales son entrenadas, una para cada predicción.

Las distintas opciones se visualizan en la figura 25.

Predicciones de múltiples variables

Este caso es muy similar al de múltiples horizontes predictivos. Cuando el objetivo es predecir múltiples variables hay dos alternativas de las expuestas anteriormente. La predicción directa, donde se generan las predicciones para cada variable en forma simultánea en una misma red neuronal, y la predicción con múltiples redes neuronales, donde para cada variable se utiliza una red neuronal distinta.

Enfoque utilizado

El enfoque que se utiliza para el presente Trabajo Final Integrador es un híbrido entre el enfoque directo y el de múltiples redes neuronales. El mismo es

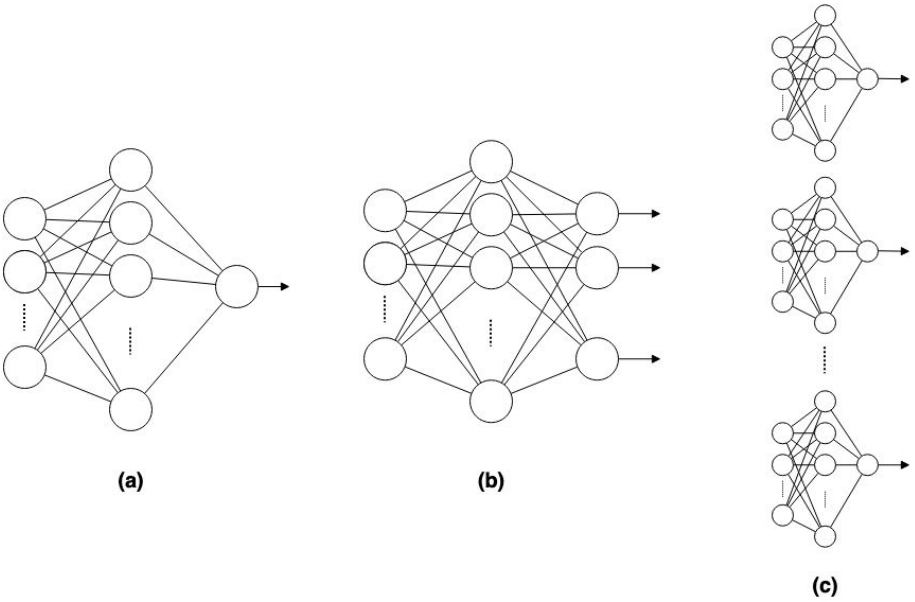

Figura 25: a)Predicción iterativa b)Predicción directa c)Predicción con múltiples redes neuronales

elegido porque el formato de sus resultados es el más parecido al arrojado por el modelo estadístico y permite además visualizar la naturaleza predictiva de cada modelo, pero podría no ser la mejor manera para obtener la mejor precisión de cada variable.

Para cada arquitectura distinta (explicadas en la siguiente sección) MLP, Conv, LSTM, LSTM+Conv y Ensamble se entrena un modelo de enfoque directo con [*pasos temporales*características*] salidas para cada horizonte predictivo (10min, 24hs, 48hs y 72hs).

- Horizonte de 10 minutos, es decir, la predicción de la próxima observación de cada variable. Para este caso los modelos entrenados tienen como salida solamente la predicción de la próxima observación de cada una de las 5 variables de la serie, 5 salidas en total.
- Horizonte de 24 horas, es decir, la predicción de las próximas 144 observaciones de cada variable. Para este caso los modelos entrenados tienen como salida la predicción de las próximas 144 observaciones de cada una de las 5 variables de la serie, $144 * 5$ salidas en total.
- Horizonte de 48 horas, es decir, la predicción de las próximas 288 observaciones de cada variable. Para este caso los modelos entrenados tienen como salida la predicción de las próximas 288 observaciones de cada una de las 5 variables de la serie, $288 * 5$ salidas en total.
- Horizonte de 72 horas, es decir, la predicción de las próximas 432 observaciones de cada variable. Para este caso los modelos entrenados tienen

como salida la predicción de las próximas 432 observaciones de cada una de las 5 variables de la serie, $432 * 5$ salidas en total.

Por consiguiente se entrenaron 20 modelos distintos del enfoque predictivo que se muestra en la figura 26.

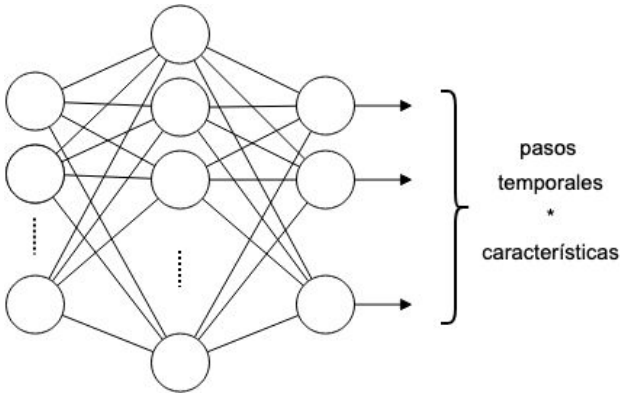

Figura 26: Enfoque predictivo utilizado.

5.3.3. Arquitecturas de los modelos

A continuación se presentan las implementaciones con Keras (ver Anexo) de las distintas arquitecturas de los modelos de aprendizaje automático, tomando como base lo presentado en [13, 5, 37]. Las distintas arquitecturas son elegidas por sus diferencias fundamentales entre sí, buscando obtener resultados diferentes a partir de cada modelo y cuyo aporte contribuya de forma positiva al modelo de ensamble.

En todos los casos se utiliza MAE como función de pérdida ya que de acuerdo a [5] es una métrica comúnmente utilizada en problemas de regresión, la cual es especificada al momento de compilar el modelo. A su vez la salida de todos los modelos es de la forma 1D compuesta por:

*[pasos temporales de salida * características de salida]*

a la que luego se le aplica una reestructuración al formato 2D [pasos temporales, features] para su interpretación.

La cantidad de neuronas $future\_target * features$ que se utiliza en las capas ocultas corresponde al número de características de la serie multiplicado por la cantidad de predicciones a obtener del modelo, el cual depende del horizonte predictivo. Por ejemplo, para el caso del horizonte predictivo de 72hs, $future\_target * features$ corresponde a $432 * 5$.

En todos los modelos se utiliza RMSProp como algoritmo de optimización. Se realizaron pruebas con distintos algoritmos, por ejemplo Adam, el cual arrojaba resultados similares. Se selecciona finalmente RMSProp por ser uno de los sugeridos para este tipo de problemas [5].

Modelo MLP - A continuación se muestra la arquitectura del modelo basado en MLP que utiliza dos capas de tipo Dense. Como las capas Dense esperan una entrada de dimensiones $[muestras, caractersticas]$, antes de entrenar el modelo es necesario aplanar el formato o estructura de los datos. Esto se logra utilizando la capa Flatten de Keras (ver Anexo). La cantidad de neuronas que componen la capa oculta de la red se obtuvo luego de un proceso experimental en el que se hicieron pruebas con distintos valores sobre un mismo grupo predictivo. Se analizaron los resultados con las siguientes cantidades:

- el mismo número de neuronas que componen la capa de entrada.
- 1, 2 y 3 veces el número de neuronas que componen la capa de salida.

Se utiliza en la capa oculta finalmente el mismo número de neuronas que en la capa de salida ya que para valores más grandes no se vieron mejoras en los resultados, y a su vez, para valores menores los resultados se vieron negativamente afectados.

$modelo = tf.keras.models.Sequential()$
$modelo.add(tf.keras.layers.Flatten(input\_shape = (past\_history, features)))$
$modelo.add(tf.keras.layers.Dense(future\_target * features,$
$activation =' relu'))$
$modelo.add(tf.keras.layers.Dense(future\_target * features))$
$modelo.compile(optimizer = tf.keras.optimizers.RMSprop(clipvalue = 1{,}0),$
$loss =' mae')$

Modelo LSTM - Este modelo se compone de dos capas LSTM apiladas y una capa Dense de salida y recibe como entrada un tensor 3D de formato [muestras, pasos temporales, características]. La cantidad de neuronas en las capas LSTM se obtuvo a partir de lo presentado en [5] para la resolución de un problema similar.

$modelo = tf.keras.models.Sequential()$
$modelo.add(tf.keras.layers.LSTM(32, return\_sequences = True,$
$input\_shape = (past\_history, features)))$
$modelo.add(tf.keras.layers.LSTM(16, activation =' relu'))$
$modelo.add(tf.keras.layers.Dense(future\_target * features))$
$modelo.compile(optimizer = tf.keras.optimizers.RMSprop(clipvalue = 1{,}0),$
$loss =' mae')$

Modelo Conv (Convolucional) - La arquitectura de este modelo consiste en una capa convolucional de filtros 1D, una de pooling 1D, una operación de aplanado, y dos capas de perceptrón multicapa que interpretan las características extraídas por la sección convolucional del modelo. Se utiliza una capa

Flatten entre las capas convolucionales y las perceptrón multicapa para reducir el mapeo de características a un vector unidimensional. La cantidad de neuronas de la capa oculta Dense y la cantidad y el tamaño de filtros de la capa Conv1D se obtuvieron a partir de un proceso experimental tomando como base lo presentado en [5, 37] para problemas similares.

$modelo = tf.keras.models.Sequential()$
$modelo.add(tf.keras.layers.Conv1D(64, 5, activation =' relu',$
$input\_shape = (past\_history, features)))$
$modelo.add(tf.keras.layers.MaxPooling1D())$
$modelo.add(tf.keras.layers.Flatten())$
$modelo.add(tf.keras.layers.Dense(50, activation =' relu'))$
$modelo.add(tf.keras.layers.Dense(features * future\_target))$
$modelo.compile(optimizer = tf.keras.optimizers.RMSprop(clipvalue = 1{,}0),$
$loss =' mae')$

Modelo Conv-LSTM - La arquitectura de este modelo híbrido combina una red neuronal convolucional con una LSTM. La red neuronal convolucional interpreta subsecuencias de la entrada que juntas son provistas como secuencia a un modelo LSTM. Es una estrategia obtenida a partir de [5, 37] que combina la velocidad de las redes neuronales convolucionales con la sensibilidad secuencial de las redes neuronales recurrentes. La red convolucional genera características de más alto nivel, luego dicha secuencia se convierte en la entrada a la sección recurrente de la red. La cantidad de neuronas en la capa LSTM y la cantidad y el tamaño de filtros de las capas Conv1D se obtuvieron a partir de las arquitecturas presentadas anteriormente.

$modelo = tf.keras.models.Sequential()$
$modelo.add(tf.keras.layers.Conv1D(32, 5, activation =' relu',$
$input\_shape = (past\_history, features)))$
$modelo.add(tf.keras.layers.MaxPooling1D(3))$
$modelo.add(tf.keras.layers.Conv1D(32, 5, activation =' relu'))$
$modelo.add(tf.keras.layers.LSTM(32, dropout = 0{,}1,$
$recurrent\_dropout = 0{,}5))$
$modelo.add(tf.keras.layers.Dense(future\_target * features))$
$modelo.compile(optimizer = tf.keras.optimizers.RMSprop(clipvalue = 1{,}0),$
$loss =' mae')$

Modelo Ensamble - El modelo de ensamble está constituido por un vector con un peso ponderado por cada predictor basado en redes neuronales y una función que toma las predicciones de dichos predictores (considerados predictores débiles) para generar una predicción ponderada promedio.

Los pesos son aprendidos a partir de los datos para entrenamiento, utilizando el algoritmo de optimización $DifferentialEvolution$ [39] de la librería *Scipy* (ver Anexo).

5.3.4. Evaluación de los modelos

Una vez que se dispone de los modelos entrenados con p pasos temporales por cada muestra de entrada, para poder realizar una predicción es necesario alimentarlos con las últimas p observaciones de datos pasados. Al igual que el modelo estadístico, los modelos basados en redes neuronales se entrenan con datos transformados. Por esta razón su dominio predictivo también está transformado. El paso siguiente a generar las predicciones es cambiar su dominio, revertir su transformación implícita. Con el objetivo de obtener mediciones sobre la precisión de las predicciones realizadas por los distintos modelos a los 10 minutos, 24 horas, 48 horas y 72 horas se entrenan 4 grupos de predictores distintos, un grupo para cada horizonte.

Para asegurar que los mismos resultados puedan reproducirse, se utiliza la función de Tensorflow (ver Anexo) tf.random.set_seed(), la cual asigna un valor fijo a las operaciones de dicha librería que inician con un valor estocástico.

5.4. Resultados

Los resultados numéricos obtenidos se muestran en los cuadros 4, 5, 6, 7 y 8. Corresponden al error absoluto medio promedio de las muestras de predicciones realizadas a lo largo de un mes consecutivo de muestras pertenecientes al conjunto de datos para testeo. A su vez, los valores están en la unidad de medida correspondiente de cada serie.

Los grupos de modelos se distinguen con los siguientes nombres:

- G1, modelos con horizonte predictivo de 72 horas.
- G2, modelos con horizonte predictivo de 48 horas.
- G3, modelos con horizonte predictivo de 24 horas.
- G4, modelos con horizonte predictivo de 10 minutos.

Cuadro 4: MAEs promedio de las predicciones de $p(mbar)$ - Presión del aire

Grupo	Modelo	10min	24hs	48hs	72hs
	MLP	0.940	5.552	7.191	7.916
	LSTM	1.056	5.156	7.252	**7.249**
G1	Conv	0.819	4.757	7.577	8.523
	Conv+LSTM	1.188	5.013	7.183	7.428
	Ensamble	0.696	4.877	6.885	7.250
	MLP	0.817	4.961	6.569	-
	LSTM	0.779	4.954	6.684	-
G2	Conv	0.555	5.878	8.384	-
	Conv+LSTM	0.890	5.083	6.733	-
	Ensamble	0.455	4.591	**6.357**	-
	MLP	0.545	4.921	-	-
	LSTM	0.537	4.262	-	-
G3	Conv	0.550	4.377	-	-
	Conv+LSTM	1.037	4.805	-	-
	Ensamble	0.347	4.365	-	-
	MLP	1.566	-	-	-
	LSTM	0.187	-	-	-
G4	Conv	0.557	-	-	-
	Conv+LSTM	0.604	-	-	-
	Ensamble	0.161	-	-	-
-	VAR	**0.050**	**4.148**	6.464	7.335

Cuadro 5: MAEs promedio de las predicciones de $T(C^o)$ - Temperatura del aire

Grupo	Modelo	10min	24hs	48hs	72hs
	MLP	1.177	3.499	4.628	5.305
	LSTM	0.812	2.987	4.458	5.292
G1	Conv	1.050	2.905	4.396	5.108
	Conv+LSTM	0.943	3.122	4.474	5.183
	Ensamble	0.519	2.958	4.264	**5.056**
	MLP	0.652	2.979	4.427	-
	LSTM	0.951	3.018	4.468	-
G2	Conv	0.756	3.239	4.797	-
	Conv + LSTM	0.749	2.990	**4.177**	-
	Ensamble	0.469	2.886	4.242	-
	MLP	0.681	3.069	-	-
	LSTM	0.578	**2.795**	-	-
G3	Conv	1.040	2.995	-	-
	Conv+LSTM	0.691	2.808	-	-
	Ensamble	0.498	2.814	-	-
	MLP	0.930	-	-	-
	LSTM	0.331	-	-	-
G4	Conv	0.713	-	-	-
	Conv+LSTM	0.373	-	-	-
	Ensamble	0.204	-	-	-
-	VAR	**0.097**	2.951	4.428	5.257

Cuadro 6: MAEs promedio de las predicciones de $rh(\%)$ - Humedad relativa

Grupo	Modelo	10min	24hs	48hs	72hs
G1	MLP	1.631	8.561	8.566	7.097
	LSTM	3.249	8.269	8.009	6.928
	Conv	2.596	7.953	8.553	7.477
	Conv+LSTM	2.725	7.677	7.886	7.119
	Ensamble	2.179	7.807	7.882	**6.777**
G2	MLP	2.081	7.519	8.144	-
	LSTM	2.584	7.963	7.972	-
	Conv	3.996	8.142	8.278	-
	Conv + LSTM	2.863	**7.346**	8.070	-
	Ensamble	1.939	7.511	**7.824**	-
G3	MLP	1.383	7.468	-	-
	LSTM	2.076	7.921	-	-
	Conv	2.349	8.149	-	-
	Conv+LSTM	2.199	7.938	-	-
	Ensamble	1.492	7.759	-	-
G4	MLP	2.682	-	-	-
	LSTM	0.597	-	-	-
	Conv	1.378	-	-	-
	Conv+LSTM	1.375	-	-	-
	Ensamble	0.619	-	-	-
-	VAR	**0.451**	8.712	8.574	7.150

Cuadro 7: MAEs promedio de las predicciones de $rho(g/m_3)$ - Densidad del aire

Grupo	Modelo	10min	24hs	48hs	72hs
G1	MLP	2.968	17.363	28.411	30.277
	LSTM	4.184	18.385	27.739	30.532
	Conv	4.950	**16.581**	28.421	32.040
	Conv+LSTM	4.837	18.764	28.214	30.156
	Ensamble	2.763	17.362	27.090	**29.698**
G2	MLP	3.767	17.827	27.962	-
	LSTM	4.764	17.905	27.049	-
	Conv	3.641	20.205	30.839	-
	Conv+LSTM	3.782	18.056	26.629	-
	Ensamble	2.308	16.788	**26.390**	-
G3	MLP	3.656	18.759	-	-
	LSTM	2.960	16.612	-	-
	Conv	5.214	17.706	-	-
	Conv + LSTM	3.420	16.949	-	-
	Ensamble	2.308	16.809	-	-
G4	MLP	2.853	-	-	-
	LSTM	1.399	-	-	-
	Conv	3.316	-	-	-
	Conv+LSTM	2.017	-	-	-
	Ensamble	0.918	-	-	-
-	VAR	**0.454**	16.991	26.510	31.639

Cuadro 8: MAEs promedio de las predicciones de $wv(m/s)$ - Velocidad del viento

Grupo	Modelo	10min	24hs	48hs	72hs
G1	MLP	0.510	1.178	1.217	1.063
	LSTM	0.715	1.140	1.153	1.075
	Conv	0.586	1.149	1.203	1.105
	Conv+LSTM	0.612	1.126	1.162	1.048
	Ensamble	0.563	1.126	1.163	**1.044**
G2	MLP	0.445	1.133	1.149	-
	LSTM	0.536	1.145	1.146	-
	Conv	0.591	1.178	1.182	-
	Conv+LSTM	0.610	1.104	**1.124**	-
	Ensamble	0.465	1.116	**1.124**	-
G3	MLP	0.417	1.100	-	-
	LSTM	0.554	**1.071**	-	-
	Conv	0.587	1.136	-	-
	Conv+LSTM	0.517	1.135	-	-
	Ensamble	0.491	1.084	-	-
G4	MLP	0.489	-	-	-
	LSTM	0.358	-	-	-
	Conv	0.501	-	-	-
	Conv+LSTM	0.414	-	-	-
	Ensamble	**0.360**	-	-	-
-	VAR	0.363	1.266	1.308	1.339

5.5. Gráficos de predicciones

Las figuras 27, 28, 29, 30, 31 y 32 corresponden a la predicción realizada sobre la variable perteneciente a la temperatura utilizando los distintos métodos predictivos del grupo con horizonte predictivo de 72 horas junto con el método estadístico sobre la misma muestra.

Sobre el eje vertical se encuentra la unidad de medida de la variable, en este caso grados Celsius (C^o). El eje horizontal de cada gráfico numera en orden ascendente cada una de las observaciones. El intervalo entre la observación nro. 0 hasta la nro. 99 corresponde a la parte de la muestra que se utiliza para realizar la predicción. A partir de la observación nro. 100, en adelante, se muestra en color naranja el valor real de la serie y en color azul el valor que predice el modelo. Dicha lógica es utilizada en todos los gráficos de predicciones de aquí en adelante.

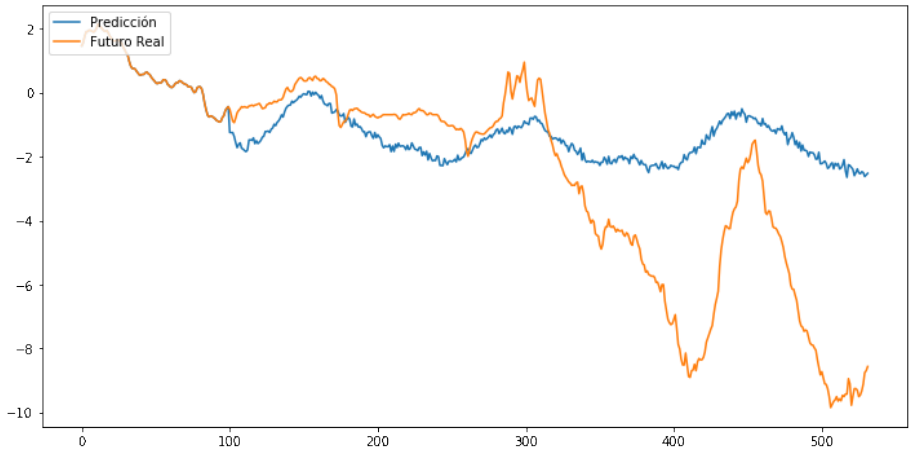

Figura 27: Predicción de la temperatura del aire $T(C^o)$ realizada por el modelo MPL de horizonte predictivo de 72 horas.

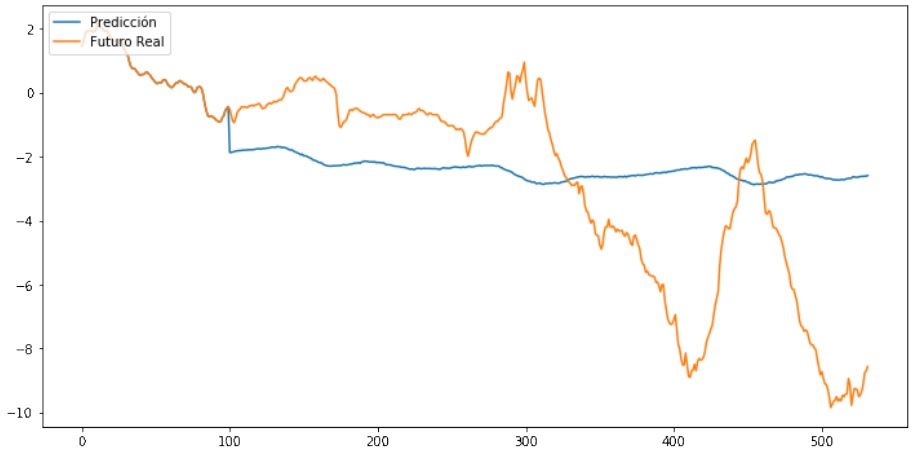

Figura 28: Predicción de la temperatura del aire $T(C^o)$ realizada por el modelo LSTM de horizonte predictivo de 72 horas.

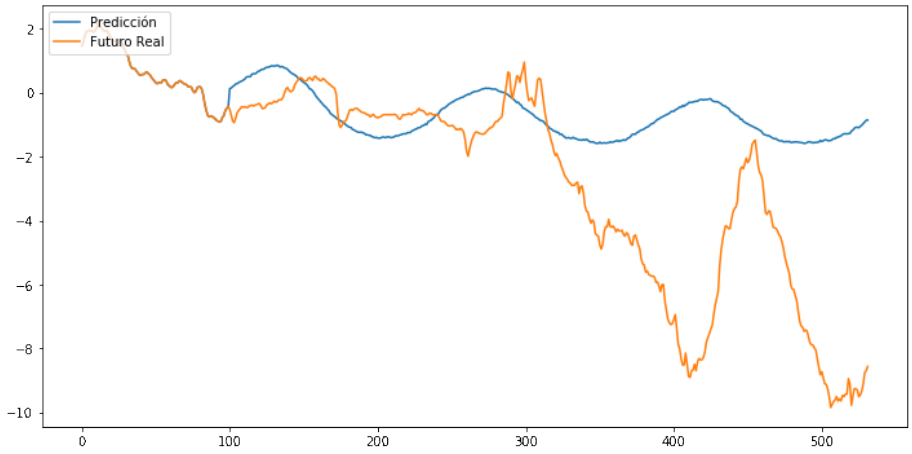

Figura 29: Predicción de la temperatura del aire $T(C^o)$ realizada por el modelo Conv de horizonte predictivo de 72 horas.

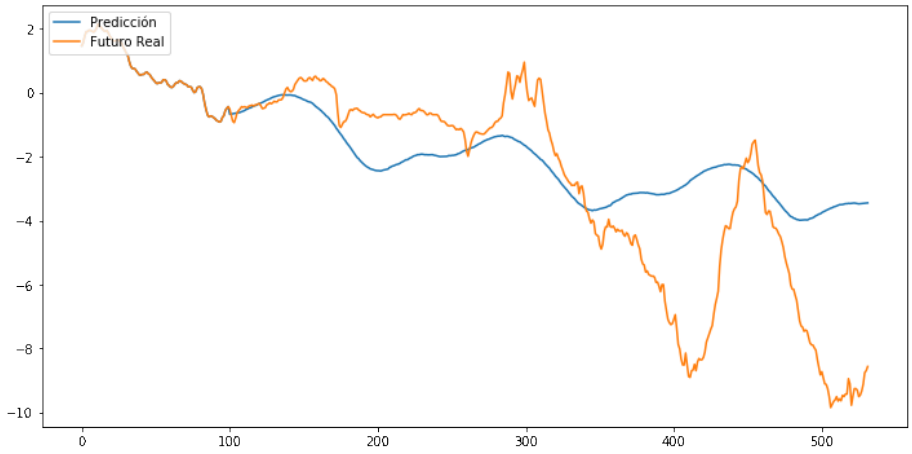

Figura 30: Predicción de la temperatura del aire $T(C^o)$ realizada por el modelo Conv+LSTM de horizonte predictivo de 72 horas.

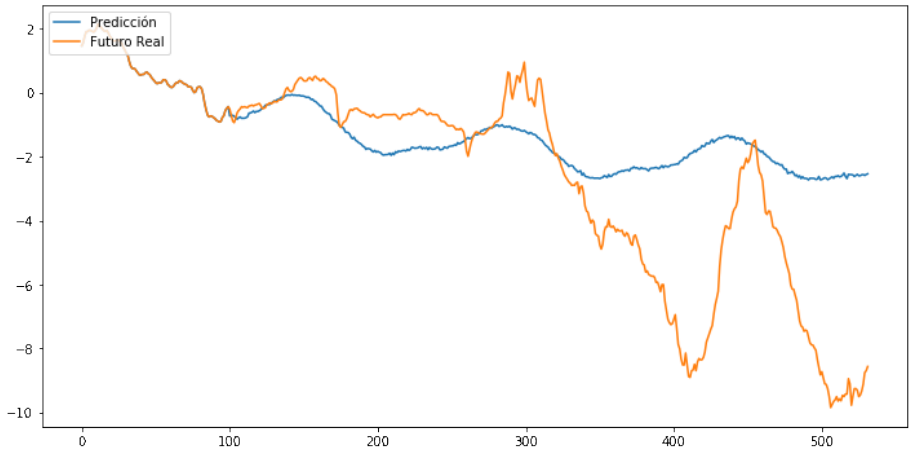

Figura 31: Predicción de la temperatura del aire $T(C^o)$ realizada por el modelo Ensamble de horizonte predictivo de 72 horas.

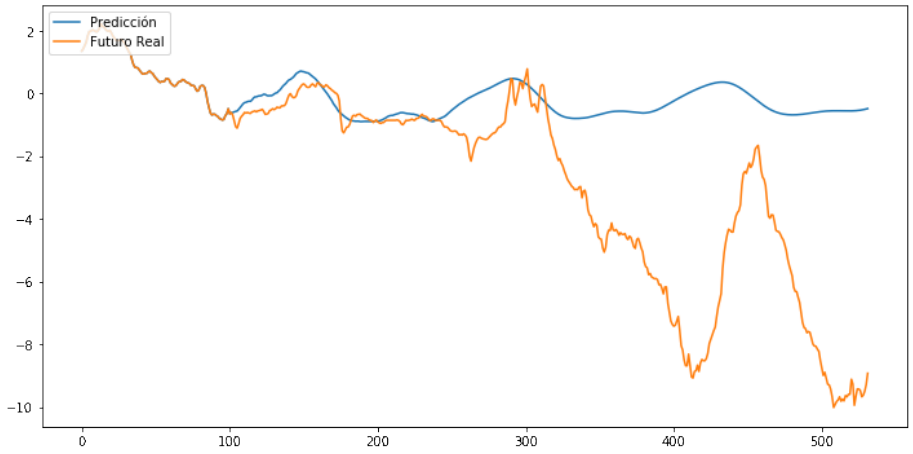

Figura 32: Predicción de la temperatura del aire $T(C^o)$ realizada por el modelo VAR de horizonte predictivo de 72 horas.

5.6. Análisis general de resultados

Para facilitar la lectura de los resultados, la figura 33 presenta un gráfico de barras de la agrupación de los resultados obtenidos de acuerdo a los distintos horizontes predictivos. Se apilan los MAE promedio resultantes de cada modelo para poder tener una representación general de cada uno.

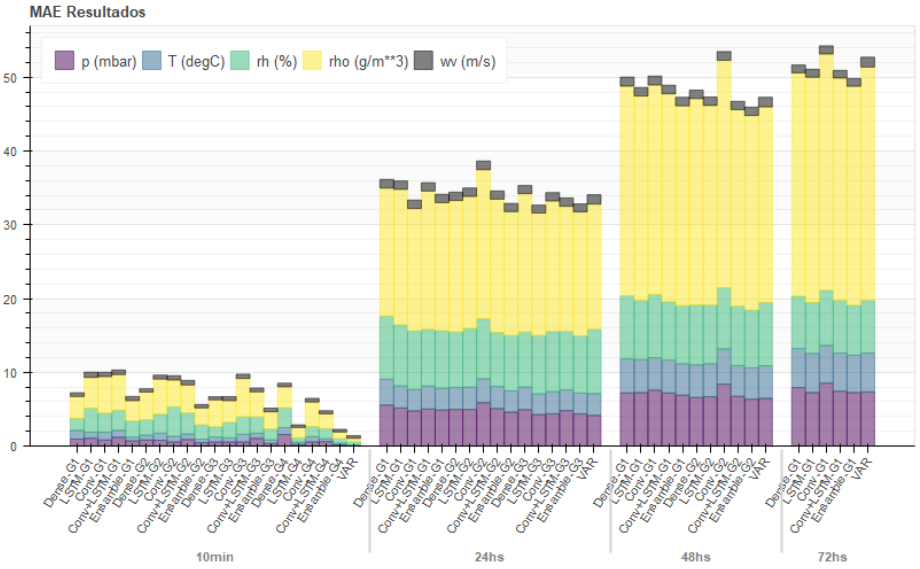

Figura 33: MAEs promedio resultantes apilados por modelo.

Las figuras 34, 35, 36, 37 y 38 presentan los gráficos de barras de los resultados de cada una de las series por separado para su mejor su comparación.

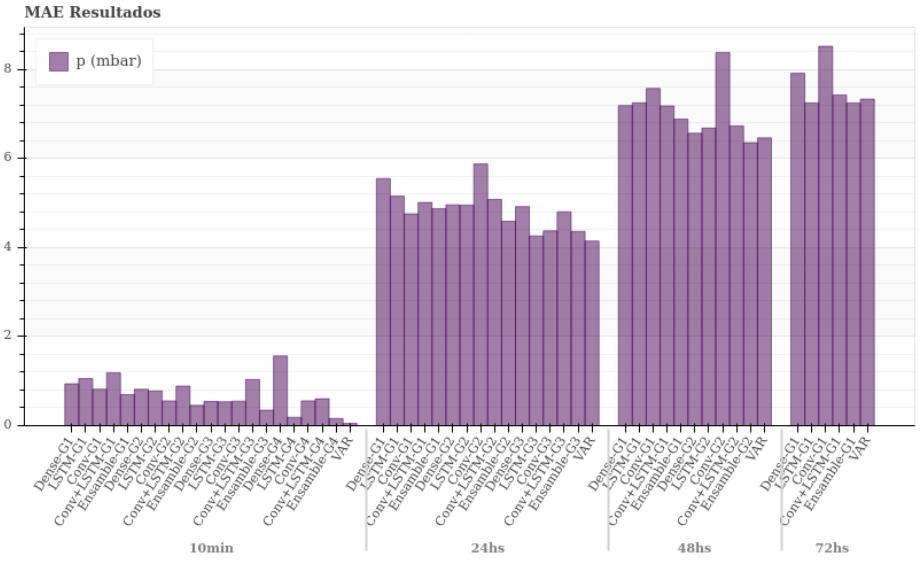

Figura 34: MAEs promedio resultantes de la variable $p(mbar)$.

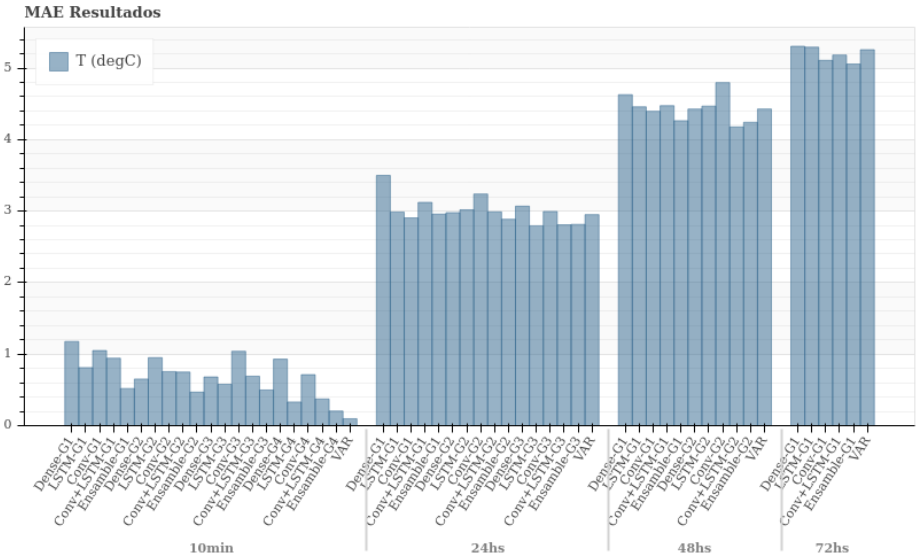

Figura 35: MAEs promedio resultantes de la variable $T(degC)$.

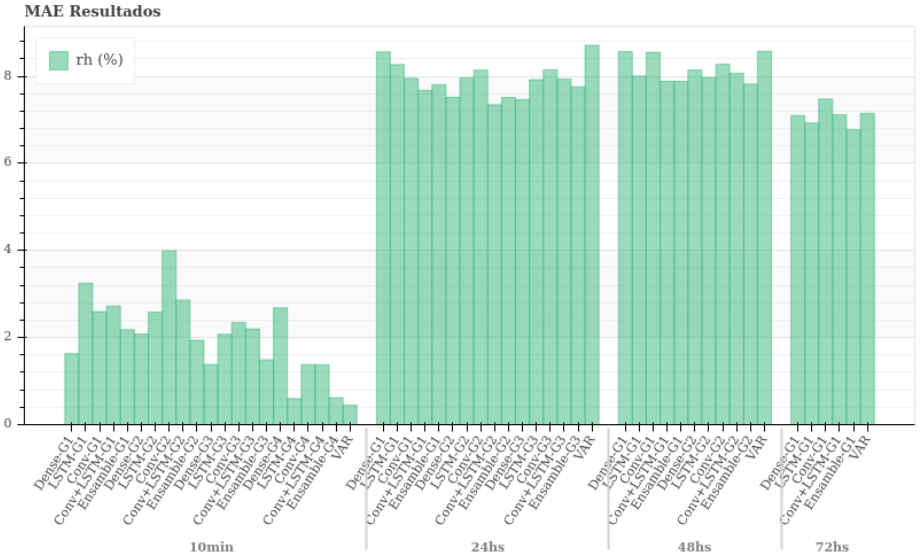

Figura 36: MAEs promedio resultantes de la variable $rh(\,\%)$.

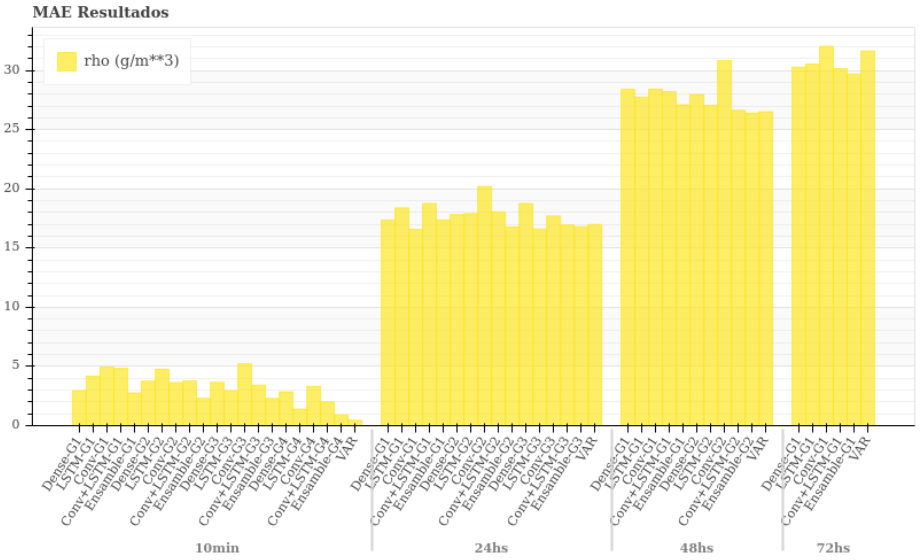

Figura 37: MAEs promedio resultantes de la variable $rho(g/m^3)$.

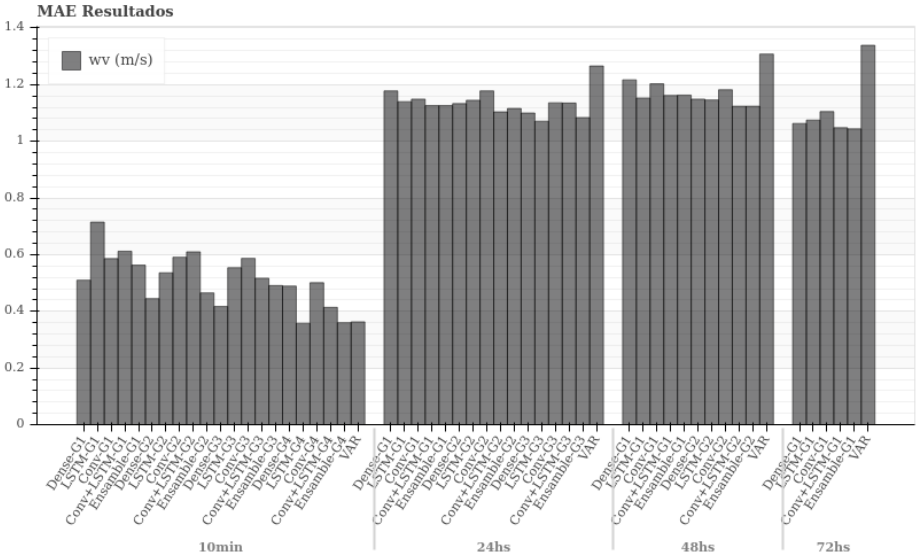

Figura 38: MAEs promedio resultantes de la variable $wv(m/s)$.

A partir de los resultados anteriores se puede observar lo siguiente:

- El ensamble (modelo 5) es el modelo más preciso de aprendizaje automático en cada grupo.
- Al aumentar el horizonte predictivo la precisión de las predicciones generalmente disminuye, sin embargo esto no es una regla. En el caso particular

de la serie wv(m/s), la precisión que muestran las predicciones obtenidas por los modelos basados en redes neuronales para el horizonte predictivo de 72hs es mejor que para el de 48hs y 24hs.

- La relación entre el horizonte predictivo y la precisión de los modelos no es lineal. Se aprecia un mayor impacto negativo en la precisión de todos los modelos al aumentar el horizonte predictivo de 10 minutos a 24 horas, que al aumentar de 24 horas a 48 horas, o de 48 horas a 72 horas. Cada aumento en el horizonte predictivo afecta la precisión, pero en menor proporción.
- Las diferencias entre los resultados del error absoluto medio de cada modelo para los horizontes predictivos de 24, 48 y 72 horas y una misma variable es menor.
- En el caso del horizonte predictivo más próximo, el de 10 minutos, el modelo Vector Autorregresivo es, en la mayoría de los casos, el más preciso. Es en este horizonte predictivo donde se observan las mayores diferencias.
- Para los casos de los horizontes predictivos de 24, 48 y 72 horas, el modelo de aprendizaje automático correspondiente al ensamble muestra mejores resultados que el modelo estadístico.
- A medida que se extiende el horizonte predictivo de un modelo de aprendizaje automático, la precisión de sus resultados se degrada en toda su extensión. Por ejemplo, los resultados de las mediciones obtenidas a 48 horas por el grupo de modelos entrenados con un horizonte predictivo de 48 horas es mejor que los resultados de las mediciones obtenidas a 48 horas por el grupo de modelos entrenados con un horizonte predictivo de 72 horas. A su vez, los resultados de las mediciones obtenidas a 24 horas por el grupo de modelos entrenados con un horizonte predictivo de 24 horas es mejor que los resultados de las mediciones obtenidas a 24 horas por el grupo de modelos entrenados con un horizonte predictivo de 48 horas.

5.7. Pruebas con información temporal

Por medio de la creación o la inclusión de variables nuevas las posibilidades de experimentación se amplían. Se realizaron pruebas en este sentido utilizando la información temporal presente en las observaciones, sobre los modelos de aprendizaje automático.

Dicha información requiere un tratamiento diferenciado para su procesamiento por medio del modelo VAR, fuera del alcance de los objetivos del presente Trabajo Final Integrador. Por esta razón no se incluyen resultados de dicho modelo.

El dato originalmente asociado a cada observación con el formato AAAA-MM-DD HH:MM:SS es desglosado en dos nuevas características 'mes' y 'día'. Se agregan a la serie para entrenamiento como variables nuevas de entrada, e ingresan a la red en formato numérico. En los cuadros 9, 10, 11, 12 y 13 se muestran los resultados obtenidos.

Cuadro 9: MAEs promedio de las predicciones de $p(mbar)$ - Presión del aire

Grupo	Modelo	10min	24hs	48hs	72hs
G1	MLP	0.611	3.998	5.97	6.268
	LSTM	1.72	4.787	7.367	7.983
	Conv	1.177	5.167	7.298	7.68
	Conv+LSTM	1.492	4.825	6.848	7.059
	Ensamble	0.606	4.078	6.011	**6.241**
G2	MLP	0.647	4.005	5.877	-
	LSTM	1.571	4.638	5.932	-
	Conv	0.821	4.954	6.194	-
	Conv+LSTM	2.77	5.581	7.406	-
	Ensamble	0.645	**3.988**	**5.770**	-
G3	MLP	0.368	4.025	-	-
	LSTM	0.814	4.522	-	-
	Conv	0.85	4.313	-	-
	Conv+LSTM	1.417	4.473	-	-
	Ensamble	0.343	4.052	-	-
G4	MLP	1.062	-	-	-
	LSTM	0.255	-	-	-
	Conv	0.742	-	-	-
	Conv+LSTM	0.69	-	-	-
	Ensamble	**0.212**	-	-	-

Cuadro 10: MAEs promedio de las predicciones de $T(C^o)$ - Temperatura del aire

Grupo	Modelo	10min	24hs	48hs	72hs
G1	MLP	0.569	3.007	4.168	4.711
	LSTM	2.286	4.147	5.337	5.171
	Conv	1.388	3.571	4.538	5.099
	Conv+LSTM	2.026	3.555	**4.069**	**4.503**
	Ensemble	0.796	3.158	4.202	4.648
G2	MLP	0.604	**2.938**	4.158	-
	LSTM	1.038	3.252	4.3	-
	Conv	1.255	3.263	4.426	-
	Conv+LSTM	2.06	3.559	4.515	-
	Ensemble	0.509	2.976	4.105	-
G3	MLP	0.54	3.073	-	-
	LSTM	0.66	3.071	-	-
	Conv	1.332	3.354	-	-
	Conv+LSTM	1.527	3.231	-	-
	Ensemble	0.555	3.053	-	-
G4	MLP	0.385	-	-	-
	LSTM	0.289	-	-	-
	Conv	0.571	-	-	-
	Conv+LSTM	0.719	-	-	-
	Ensemble	**0.262**	-	-	-

Cuadro 11: MAEs promedio de las predicciones de $rh(\%)$ - Humedad relativa

Grupo	Modelo	10min	24hs	48hs	72hs
G1	MLP	2.406	7.723	9.124	9.16
	LSTM	5.948	8.739	8.779	8.524
	Conv	3.35	8.314	9.229	8.104
	Conv+LSTM	4.837	**7.077**	8.086	**7.384**
	Ensamble	1.968	7.442	8.571	8.378
G2	MLP	1.891	7.906	8.833	-
	LSTM	4.029	8.741	9.192	-
	Conv	3.747	7.969	8.6	-
	Conv+LSTM	3.841	7.65	**7.916**	-
	Ensamble	1.824	7.9	8.749	-
G3	MLP	1.524	8.076	-	-
	LSTM	2.598	8.511	-	-
	Conv	3.56	8.323	-	-
	Conv+LSTM	3.168	7.732	-	-
	Ensamble	1.429	7.986	-	-
G4	MLP	1.825	-	-	-
	LSTM	**0.694**	-	-	-
	Conv	1.84	-	-	-
	Conv+LSTM	1.388	-	-	-
	Ensamble	0.705	-	-	-

Cuadro 12: MAEs promedio de las predicciones de $rho(g/m_3)$ - Densidad del aire

Grupo	Modelo	10min	24hs	48hs	72hs
G1	MLP	2.761	17.329	24.69	27.492
	LSTM	10.773	25.0	32.644	33.139
	Conv	7.067	20.973	27.373	29.444
	Conv+LSTM	9.657	21.118	26.447	28.441
	Ensamble	3.977	18.264	25.193	**27.424**
G2	MLP	2.442	16.876	24.659	-
	LSTM	5.142	19.27	25.319	-
	Conv	5.682	19.549	27.387	-
	Conv+LSTM	9.936	22.186	29.888	-
	Ensamble	2.445	17.204	**24.527**	-
G3	MLP	2.446	**17.064**	-	-
	LSTM	3.676	18.944	-	-
	Conv	6.395	19.202	-	-
	Conv+LSTM	7.894	19.02	-	-
	Ensamble	2.791	17.543	-	-
G4	MLP	2.461	-	-	-
	LSTM	1.832	-	-	-
	Conv	2.811	-	-	-
	Conv+LSTM	3.382	-	-	-
	Ensamble	**1.514**	-	-	-

Cuadro 13: MAEs promedio de las predicciones de $wv(m/s)$ - Velocidad del viento

Grupo	Modelo	10min	24hs	48hs	72hs
G1	MLP	0.486	1.115	1.104	1.047
	LSTM	0.735	1.144	1.188	1.091
	Conv	0.755	1.216	1.211	1.085
	Conv+LSTM	0.791	1.125	1.139	1.097
	Ensamble	0.477	1.111	1.104	**1.041**
G2	MLP	0.454	1.101	**1.102**	-
	LSTM	0.652	1.145	1.15	-
	Conv	0.676	1.085	1.118	-
	Conv+LSTM	0.804	1.149	1.157	-
	Ensamble	0.462	1.097	1.104	-
G3	MLP	0.435	1.072	-	-
	LSTM	0.499	1.091	-	-
	Conv	0.683	1.156	-	-
	Conv+LSTM	0.606	1.075	-	-
	Ensamble	0.428	**1.067**	-	-
G4	MLP	0.454	-	-	-
	LSTM	**0.358**	-	-	-
	Conv	0.494	-	-	-
	Conv+LSTM	0.41	-	-	-
	Ensamble	0.364	-	-	-

Se observa que se mejoran un poco las predicciones de los horizontes de 48 y 72 horas al contrario de las predicciones a 10 minutos. Estas se ven afectadas negativamente por la inclusión de las nuevas variables. Esto puede, o no, ser un aporte deseado para la predicción de una variable en particular, dependerá del objetivo del problema a resolver.

Las figuras 39, 40, 41 y 42 presentan, a modo comparativo los resultados de los modelos para los distintos horizontes predictivos. Del lado izquierdo de cada gráfico se muestran los resultados de los modelos originales, y del lado derecho los resultados obtenidos con los modelos entrenados con las nuevas variables con información temporal (+ info). La lógica es la misma que en el caso de los gráficos de barra anteriormente mostrados.

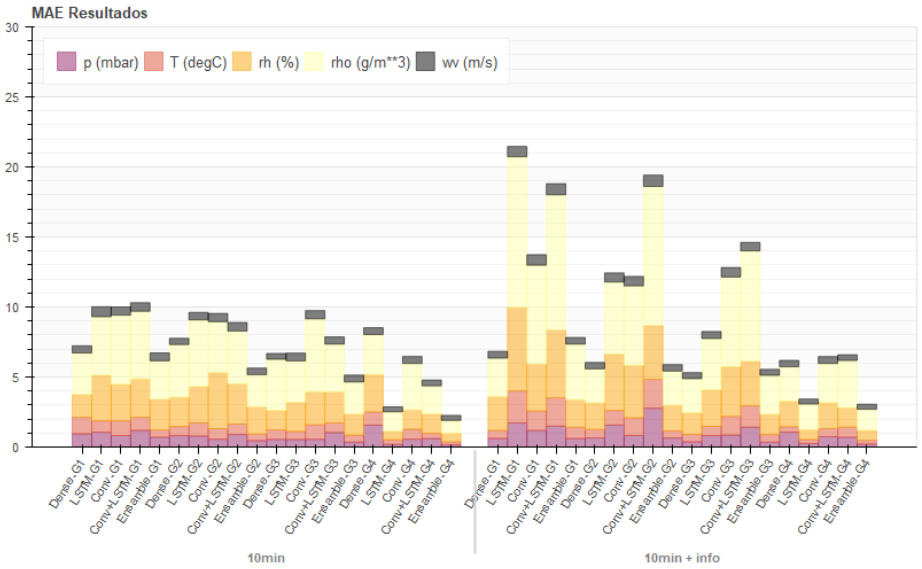

Figura 39: A izq. resultados de los modelos originales, a der. resultados con los modelos entrenados con variables con información temporal (+ info).

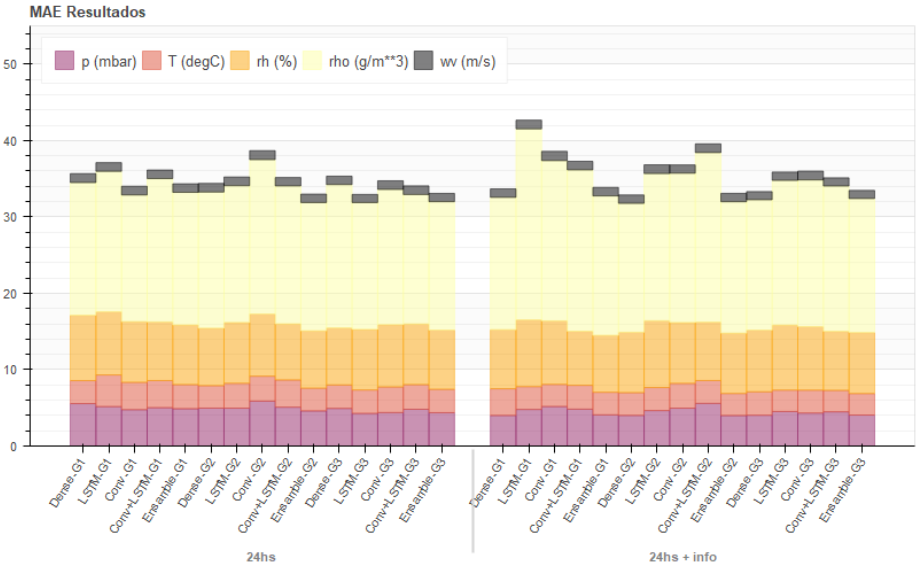

Figura 40: A izq. resultados de los modelos originales, a der. resultados con los modelos entrenados con variables con información temporal (+ info).

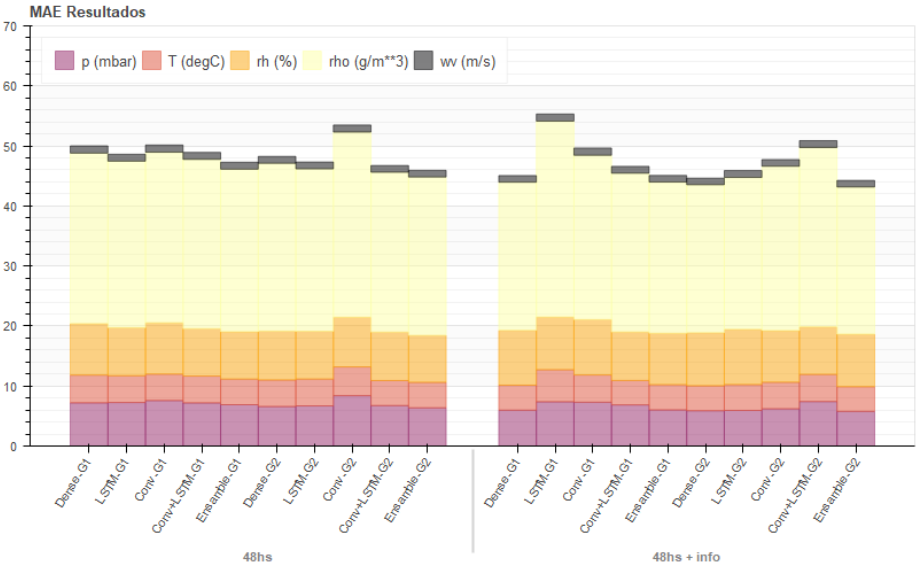

Figura 41: A izq. resultados de los modelos originales, a der. resultados con los modelos entrenados con variables con información temporal (+ info)

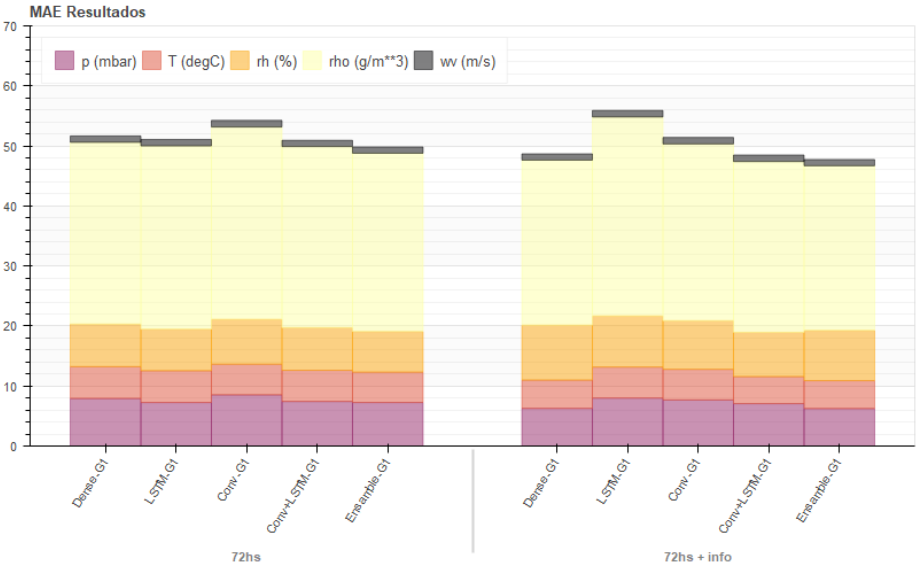

Figura 42: A izq. resultados de los modelos originales, a der. resultados con los modelos entrenados con variables con información temporal (+ info)

Las figuras 43 y 44 corresponden a las predicciones de distintos modelos con la misma arquitectura entrenados con, y sin, variables temporales sobre la misma muestra.

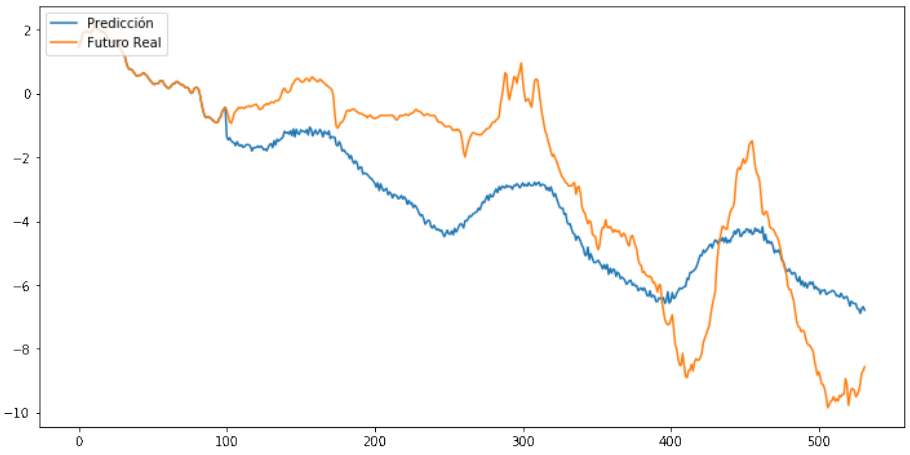

Figura 43: Predicción de la variable de la temperatura del aire $T(C^o)$ del modelo Ensamble entrenado con variables con información temporal.

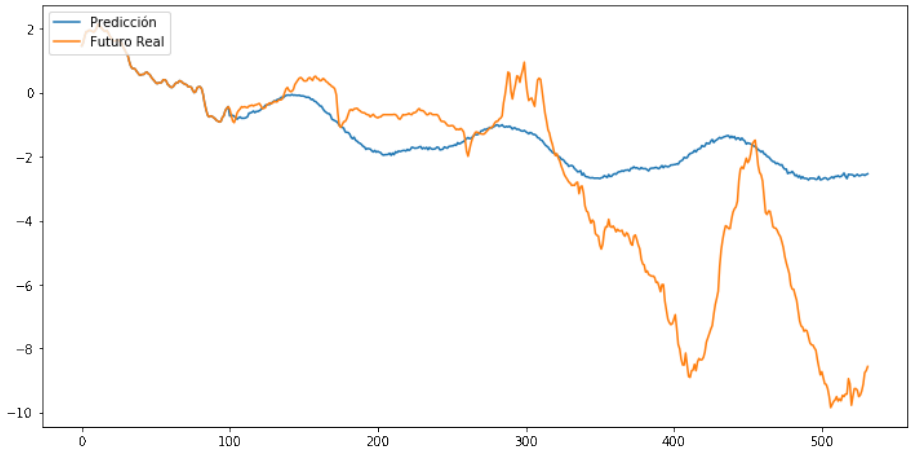

Figura 44: Predicción de la variable de la temperatura del aire $T(C^o)$ del modelo Ensamble original.

6. Conclusiones y trabajos futuros

Es destacable la simplicidad computacional del modelo estadístico frente a las redes neuronales. Esto puede ser un factor decisivo al momento de seleccionar un método predictivo para un determinado problema. El tiempo que requiere el ajuste del modelo estadístico es solo una fracción del tiempo que requiere el entrenamiento de solo uno de los modelos de aprendizaje automático.

A partir de la experiencia adquirida al realizar el presente Trabajo Final Integrador, la opinión del autor respecto a las diferencias y contradicciones entre las publicaciones que comparan los diferentes métodos predictivos, es que surgen a raíz de las diferencias fundamentales entre ambos enfoques. La cantidad de variantes para la preparación de los datos y de los modelos hace de una comparación exhaustiva o definitiva una tarea difícil. Para abordar una comparación es más indicado partir desde un contexto o marco detallado, pero esto es algo que no siempre está presente en los artículos.

Si bien hay importantes diferencias en los tiempos de ajuste o entrenamiento de los distintos métodos, es el procesamiento de los datos (análisis, transformaciones, reestructuraciones y experimentaciones) la parte más complicada del desarrollo y la que demanda la mayor parte del tiempo.

A parte de los resultados numéricos es importante tener en cuenta los resultados visuales. Los gráficos revelan en cierta forma la naturaleza de cada predictor. En este sentido, los patrones predictivos del modelo estadístico parecen suavizarse o desaparecer a medida que se extiende el horizonte. En el caso de los modelos de aprendizaje automático esto no se observa, los patrones que aprenden los modelos se mantienen. Esto puede explicar intuitivamente el motivo por el cual los modelos de aprendizaje automático responden mejor para horizontes mayores. Por otra parte, es posible apreciar a partir de las imágenes que el modelo Conv capta con mayor claridad los patrones cíclicos que el modelo MLP. Esto puede, o no, ser de utilidad. Es posible que este conocimiento pueda ser aplicado para deducir las fortalezas y debilidades de cada modelo para los datos utilizados, y determinr si un predictor realiza un aporte favorable para un ensamble.

Muchos son los factores que influyen sobre los resultados finales de los modelos predictivos, y las transformaciones en el procesamiento de datos es uno de ellos. Sobre este punto hay lugar para experimentar y obtener mejoras. Para el desarrollo del presente Trabajo Final Integrador se realizaron pruebas sobre la predicción de series temporales transformadas a estacionarias utilizando modelos de aprendizaje automático, entrenados con los mismos datos utilizados para el método estadístico. Si bien no se incluyen los resultados, éstos no fueron buenos. El aprendizaje de las redes se vió muy comprometido. Esto no significa que transformar los datos a estacionarios sea siempre inapropiado. Este es un camino con una gran cantidad de variantes, y su éxito o fracaso depende también de los datos a tratar.

Los distintos modelos no responden de la misma forma a todos los datos. Como se muestra en los resultados, ningún modelo predictivo independiente supera al resto en todas las predicciónes (el predictor Ensamble en este caso no se considera independiente). A partir de esto surge la importancia de realizar pruebas sobre una serie de datos con variables heterogéneas entre sí y el valor de los modelos que ensamblan distintos predictores.

La inclusión de nuevas variables o características tiene influencia sobre los resultados que para poder catalogarlos como positivos o negativos es necesario tener en cuenta el objetivo de las predicciones. Curiosamente los resultados que se obtienen al incluir las variables con información temporal se muestran consistentemente mejorados en las predicciones con horizontes más lejanos y consistentemente perjudicados en el horizonte más próximo a las observaciones de entrada. Esto refuerza la importancia de establecer un objetivo predictivo puntual.

Se observa que las principales diferencias entre los modelos predictivos se encuentran en el horizonte predictivo más cercano. El modelo VAR es claramente más preciso que el resto, seguido en segundo lugar por el modelo de ensamble. Al aumentar el horizonte dicha diferencia entre modelos disminuye, y pasan a tener mayor preponderancia los modelos de aprendizaje automático. En cuanto a la comparación entre modelos, no es posible simplemente determinar un ganador, o un mejor o peor modelo. Es más apropiado conocer las fortalezas de cada modelo y determinar cuál utilizar en función del problema a resolver. Si para el problema a resolver son críticas las predicciones a corto plazo, y el tiempo o los recursos disponibles no permiten el desarrollo de más de un modelo, entonces probablemente el modelo estadístico sea la mejor opción. Si el problema necesita principalmente predicciones con horizontes más extendidos, un modelo de aprendizaje automático sería un buen camino a seguir. Es importante tener en cuenta que el mejor modelo predictor surgirá de la combinación de ambos tipos de predictores, esto es, de un ensamble que combine modelos estadísticos y de aprendizaje automático.

Se puede afirmar que la predicción de series temporales mediante la utilización de redes neuronales devuelve resultados muy positivos e inclusive en determinadas circunstancias mejores que los de un método estadístico conocido por su buen desempeño. Esto significa que estos modelos son realmente útiles para aplicaciones predictivas. Sin embargo es necesario mejorarlos mediante enfoques especiales o nuevas experimentaciones para compensar la elevada carga computacional que requiere su entrenamiento.

Como posibles trabajos a futuro, aumentar la cantidad de observaciones por variable sin dudas mejoraría los modelos de aprendizaje automático, no tanto así a los estadísticos. Esto les permitiría 'ver' todavía más casos distintos al entrenar y generar de esta forma una mejor generalización de los mismos. Una manera complementaria de mejorar los resultados sería agregando modelos nuevos, que interpreten los datos de forma diferente. Hay una gran cantidad de modelos estadísticos, y de alternativas de aprendizaje automático diferentes y

de estrategias de generación de ensambles que podrían obtener y potenciar lo mejor de cada uno.

Otro aspecto pendiente es la validación de los resultados y conclusiones obtenidas utilizando conjuntos de datos diferentes. Sería esperable que para datos relacionados con fenómenos meteorológicos en un entorno real de otra zona geográfica se obtengan resultados similares. A su vez, para datos procedentes de un origen distinto, por ejemplo de la lectura de sensores de maquinaria industrial, se podría obtener una nueva percepción sobre cada modelo.

A. Anexo - Software utilizado

A continuación se mencionan los principales componentes de software utilizados en el presente trabajo.

Python es un lenguaje de programación de código abierto orientado a objetos interpretado de alto nivel con semántica dinámica, estructuras de datos de alto nivel, combinadas con tipeo dinámico y enlace dinámico. Es un lenguaje de scripting que puede ser utilizado para generar aplicaciones rápidamente. Otra de sus principales características es la facilidad de integración con librerías en otros lenguajes como 'C', lo cual lo convierten en un lenguaje de unión de componentes previamente existentes. Gracias a estas características Python tiene hoy una amplia comunidad siendo uno de los lenguajes más populares junto con otros como 'R' para el análisis de datos, computación interactiva y visualización de datos. Como desventaja, al ser Python un lenguaje de programación interpretado, su código correrá de forma más lenta que en un lenguaje compilado como C++.

NumPy es una librería dentro del ecosistema de Python utilizada para el cómputo numérico. Proveé estructuras de datos y algoritmos que son utilizadas en la mayoría de las aplicaciones científicas gracias a su procesamiento rápido y eficiente.

Pandas es otra de las principales librerías dentro del ecosistema de Python que proveé estructuras de datos de alto nivel y funciones para el análisis y estadística de información con la flexibilidad de lenguajes como SQL.

Matplotlib es una librería utilizada para la generación de visualizaciones de datos en dos dimensiones muy utilizada también en aplicaciones de ciencias de datos.

Statsmodels es una librería que contiene, entre otras cosas, algoritmos y funciones de visualización para el análisis económico y estadístico. En este trabajo se utilizan de esta librería el modelo estadístico VAR y los gráficos de autocorrelación.

Scipy es una librería compuesta de herramientas y algoritmos matemáticos. SciPy contiene módulos para optimización, álgebra lineal, integración, interpolación, funciones especiales, procesamiento de señales y otras tareas para la ciencia e ingeniería.

TensorFlow es una poderosa plataforma de código abierto desarrollada por Google para el cómputo numérico rápido y eficiente que contiene un amplio ecosistema de herramientas y librerías para aprendizaje automático y la creación y entrenamiento de redes neuronales. Tensorflow tiene soporte para su ejecución en paralelo a través de múltiples CPUs o GPUs y distribuida a través de distintos servidores. Esto permite el entrenamiento de redes de gran tamaño sobre grandes conjuntos de datos.

Keras es una interfaz de programación de aplicaciones (API, por su sigla en inglés) de alto nivel para la creación y el entrenamiento de modelos de aprendizaje automático que ofrece ventajas como una interfaz simple y la creación de modelos en forma modular. Estableciendo a Keras como una API de alto nivel para TensorFlow se logra simplificar el workflow y minimizar el desarrollo de modelos.

Jupyter Notebook es un entorno de trabajo interactivo que permite desarrollar código en Python de manera dinámica, a la vez que integrar en un mismo documento tanto bloques de código como texto, gráficos o imágenes.

7. Referencias

1. Hamid S.A. and Habib A. (2014). Financial forecasting with neural networks. Academy of Accounting and Financial Studies Journal, 18, 37-55.

2. Wang Jie and Wang Jun. (2017). Forecasting stochastic neural network based on financial empirical mode decomposition. Neural Networks, 90, 8-20.

3. Qiu M. and Song Y. (2016). Predicting the Direction of Stock Market Index Movement Using an Optimized Artificial Neural Network Model. PLoS ONE, 11(5).

4. Kock A. B. and Teräsvirta T. (2016). Forecasting macroeconomic variables using neural network models and three automated model selection techniques. Econometric Reviews, 35(8-10), 1753-1779.

5. F. Chollet. (2018). Deep Learning With Python. Shelter Island, NY, USA, Manning.

6. Zhao K. and Wang C. (2017). Sales Forecast in E-commerce using Convolutional Neural Network. ArXiv, abs/1708.07946.

7. Lipton Z.C., Kale D.C., Elkan C. and Wetzel R. (2016). Learning to Diagnose with LSTM Recurrent Neural Networks. CoRR, abs/1511.03677.

8. Bui C., Pham N., Vo A., Tran, A., Nguyen A. and Le T. (2017). Time Series Forecasting for Healthcare Diagnosis and Prognostics with the Focus on Cardiovascular Diseases. International Conference on the Development of Biomedical Engineering in Vietnam, Springer, Singapore, 63, 809-818.

9. Monidipa Das, Soumya K. Ghosh. (2018). Data-driven approaches for meteorological time series prediction: A comparative study of the state-of-the-art computational intelligence techniques. Pattern Recognition Letters, 105, 155-164.

10. Katris Christos and Daskalaki Sophia. (2014). Prediction of Internet traffic using time series and neural networks. Conference: International workconference on Time Series, 1, 594-605.

11. Gevaert Wouter, Tsenov Georgi and Mladenov Valeri. (2010). Neural networks used for speech recognition. Journal of Automatic Control, University of Belgrade, 20, 1-7.

12. Zhang Peter, Patuwo Eddy and Hu Michael. (1998). Forecasting With Artificial Neural Networks: The State of the Art. International Journal of Forecasting, 14, 35-62.

13. Makridakis Spyros, Spiliotis Evangelos and Assimakopoulos Vassilis. (2018). Statistical and Machine Learning forecasting methods: Concerns and ways forward. PLoS ONE, 13.

14. Makridakis Spyros, Spiliotis Evangelos and Assimakopoulos Vassilis. (2018). The M4 Competition: Results, findings, conclusion and way forward. International Journal of Forecasting, 34, 802-808.

15. Fry Chris, Brundage Michael. (2019). The M4 forecasting competition - A practitioner's view. International Journal of Forecasting, 36, 5.

16. Redd A., Khin K. and Marini A. (2019). Fast ES-RNN: A GPU Implementation of the ES-RNN Algorithm. ArXiv, abs/1907.03329.

17. M. Casey Brace, J. Schmidt and M. Hadlin. (1991). Comparison of the forecasting accuracy of neural networks with other established techniques. Proceedings of the First International Forum on Applications of Neural Networks to Power Systems, Seattle, WA, USA, 31-35.

18. Chakraborty Kanad, Mehrotra Kishan, Mohan Chilukuri and Ranka Sanjay. (1992). Forecasting the Behavior of Multivariate Time Series Using Neural Networks. Neural Networks, 5, 961-970.

19. Foster W.R., Collopy Fred and Ungar Lyle. (1992). Neural network forecasting of short, noisy time series. Computers and Chemical Engineering, 16, 293-297.

20. Hann Tae and Steurer Elmar. (1996). Much ado about nothing? Exchange rate forecasting: Neural networks vs. linear models using monthly and weekly data. Neurocomputing, 10, 323-339.

21. Kang S. (1991). An investigation of the use of feedforward neural networks for forecasting. Ph.D. thesis, Kent State University.

22. Maqsood Imran, Khan Muhammad and Abraham Ajith. (2004). An ensemble of neural networks for weather forecasting. Neural Computing and Applications, 13, 112-122.

23. Y. Liu, M. C. Roberts and R. Sioshansi. (2018). A vector autoregression weather model for electricity supply and demand modeling. Journal of Modern Power Systems and Clean Energy, 6, 763-776.

24. Hochreiter Sepp and Schmidhuber Jürgen. (1997). Long Short-term Memory. Neural computation, 9, 1735-1780.

25. Denis Kwiatkowski, Peter C.B. Phillips, Peter Schmidt and Yongcheol Shin. (1992). Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root?. Journal of Econometrics, 54 (1-3), 159-178.

26. Bates John and Granger Clive. (1969). The Combination of Forecasts. Journal of the Operational Research Society, 20(4), 451-468.

27. Hyndman R. J., and Athanasopoulos G. (2018). Forecasting: Principles and Practice. (2nd ed.) OTexts. https://otexts.org/fpp2/

28. Granger C. (1969). Investigating Causal Relations by Econometric Models and Cross-spectral Methods. Econometrica, 37(3), 424-438.

29. Olaf Kolle. (2008). Documentation of the Weather Station on Top of the Roof of the Institute Building of the Max-Planck-Institute of Biogeochemistry. Recuperado de https://www.bgc-jena.mpg.de/wetter.

30. Bengio Y., Simard Patrice and Frasconi Paolo. (1994). Learning long-term dependencies with gradient descent is difficult. IEEE transactions on neural networks, 5, 157-166.

31. Hochreiter Sepp and Schmidhuber Jürgen. (1997). Long Short-term Memory. Neural computation, 9, 1735-1780.

32. Aileen Nielsen. (2019). Practical Time Series Analysis: Prediction with Statistics and Machine Learning. O'Reilly.

33. Fulcher Ben. (2018). Feature-Based Time-Series Analysis. CRC Press.

34. Wang Xiaoyue, Ding Hui, Trajcevski Goce, Scheuermann Peter and Keogh Eamonn. (2013). Experimental Comparison of Representation Methods and Distance Measures for Time Series Data. Data Mining and Knowledge Discovery, 26, 275–309.

35. Brockwell P. J. and Davis R. A. (2016). Introduction to time series and forecasting (3rd ed). New York, USA, Springer.

36. Sims Christopher A. (1980). Macroeconomics and Reality. Econometrica, Econometric Society, 48, 1-48.

37. Jason Brownlee. (2017). Long Short-Term Memory Networks With Python. Machine Learning Mastery.

38. Tieleman T. and Hinton G. (2012). Lecture 6.5-rmsprop: Divide the Gradient by a Running Average of Its Recent Magnitude. Coursera: Neural Networks for Machine Learning, 4, 26-31.

39. Qin Kai, Huang Vicky and Suganthan Ponnuthurai. (2009). Differential Evolution Algorithm With Strategy Adaptation for Global Numerical Optimization. Evolutionary Computation, IEEE Transactions, 13, 398-417.

40. Holt Charles. (2004). Forecasting Seasonals and Trends by Exponential Weighted Moving Averages. International Journal of Forecasting, 20, 5-10.

41. Box G.E.P. and Cox D.R. (1964). An Analysis of Transformations. Journal of the Royal Statistical Society: Series B (Methodological), 26, 211-243.